



Arm[®] Mali[™] GPU

Version 3.5

Virtualization Guide

Non-Confidential

Copyright © 2020–2025 Arm Limited (or its affiliates).
All rights reserved.

Issue 20

102104_0305_20_en



Arm® Mali™ GPU Virtualization Guide

This document is Non-Confidential.

Copyright © 2020–2025 Arm Limited (or its affiliates). All rights reserved.

This document is protected by copyright and other intellectual property rights.

Arm only permits use of this document if you have reviewed and accepted [Arm's Proprietary Notice](#) found at the end of this document.

This document (102104_0305_20_en) was issued on 2025-04-17. There might be a later issue at <https://developer.arm.com/documentation/102104>

The product version is 3.5.

See also: [Proprietary Notice](#) | [Product and document information](#) | [Useful resources](#)

Start reading

If you prefer, you can skip to [the start of the content](#).

Intended audience

This guide is written for software developers who are developing the software stack for virtualization on top of *Graphics Processing Units* (GPUs). Typically, the reader is integrating OSes and drivers for a particular platform. This guide provides guidance on how to integrate GPUs into a virtualized platform. It describes the reference code that is used to do this integration.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

Previous issues of this document included language that can be offensive. We have replaced this language. See [Revision history](#) on page 374.

To report offensive language in this document, email terms@arm.com.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on <https://support.developer.arm.com>.

To provide feedback on the document, fill the following survey: <https://developer.arm.com/documentation-feedback-survey>.

Contents

1. Virtualization with GPUs.....	6
1.1 Mali GPU virtualization solutions.....	7
1.1.1 Paravirtualization with Mali™ GPUs.....	7
1.1.2 Hardware virtualization.....	9
1.2 Cooperative time slicing.....	10
1.2.1 Mali DDK data-flow.....	11
1.2.2 GPU yield process.....	12
1.2.3 API usage and yielding.....	13
1.2.4 Tiler queue size.....	14
1.2.5 Choosing the time slice length.....	15
2. Reference software stacks for virtualization.....	16
2.1 Overview.....	16
2.1.1 Arbiter.....	16
2.1.2 Cooperative driver.....	17
2.1.3 Communication.....	18
2.1.4 Switching the GPU.....	19
2.1.5 Power management.....	19
2.2 Paravirtualization.....	20
2.2.1 Paravirtualization time slice metrics.....	22
2.3 Hardware virtualization.....	23
2.4 Hardware separation.....	24
2.4.1 Partition management.....	26
2.5 Resource assignment with the partition manager.....	27
2.5.1 Arbiter messaging in the partition manager.....	30
2.6 Example use-cases.....	30
2.7 Reference software stack for paravirtualization.....	35
2.8 Reference software stack for hardware virtualization.....	44
2.8.1 Porting information for other operating systems.....	55
2.9 Power management in the virtualization reference stack.....	61
2.10 Build reference software for virtualization.....	68
2.11 Software debug features with paravirtualization.....	72

2.12 Protected content with paravirtualization.....	76
2.13 Software security considerations for virtualization.....	81
3. Message protocol.....	85
3.1 Protocol versioning and backward compatibility support.....	86
3.2 Protocol message format.....	88
3.2.1 Messages from arbiter to VM.....	88
3.2.2 Messages from VM to arbiter.....	90
3.3 Messaging protocol processes with example message sequences.....	92
3.3.1 Initialization message protocol.....	92
3.3.2 GPU yield.....	98
3.3.3 GPU lost.....	99
A. Limitations on virtualization in GPUs.....	100
B. Xen hypervisor.....	102
B.1 Paravirtualization worked example.....	103
B.2 Hardware virtualization worked example.....	105
B.2.1 Example System Deployment.....	105
B.2.2 Device Tree.....	105
B.2.3 Flows.....	112
C. Partition manager.....	114
C.1 Register page PARTITION_MANAGER.....	114
C.1.1 Register sub-page PTM_ACCESS_WINDOW.....	118
C.1.2 Register sub-page PTM_ASSIGN.....	129
C.1.3 Register sub-page PTM_AW_GPU.....	147
C.1.4 Register sub-page PTM_BIST_CONTROL.....	147
C.1.5 Register sub-page PTM_MESSAGE.....	147
C.1.6 Register sub-page PTM_PARTITION_CONFIG.....	150
C.1.7 Register sub-page PTM_PARTITION_CONTROL.....	164
C.1.8 Register sub-page PTM_RESOURCE_GROUP.....	189
C.1.9 Register sub-page PTM_SYSTEM.....	245
C.1.10 Access states.....	368
Proprietary Notice.....	372
Product and document information.....	374

Product status.....374

Revision history..... 374

Conventions.....378

Useful resources..... 380

1. Virtualization with GPUs

Graphics Processing Units (GPUs) can be used in virtualized systems to support *Virtual Machines* (VMs) and guest software.

Virtualization allows you to create multiple *Virtual Machines* (VMs), which share some hardware but are isolated from each other. Each VM runs a full OS with kernel and user space.

An important feature of virtualization is that VMs are isolated from one another. There are several reasons why this isolation can be useful. Examples include:

- To separate safety-critical work from non safety-critical work.
- To separate secure work from non-secure work.

You can also use virtualization as a hardware abstraction, for example to run code on a processor for which it was not originally written.

A software hypervisor, for example Xen, controls the VMs. The hypervisor runs as the system host and creates each VM as a guest with its own operating system and virtual hardware. The hypervisor is also known as a *Virtual Machine Monitor* (VMM),

This guide contains a brief overview of the different mechanisms for virtualizing a GPU. The focus is on the methods used to virtualize Mali™ GPUs.

There are several ways for VMs to share GPU hardware in a virtualized environment without costing the significant performance of GPU hardware emulation. Most approaches have VMs that are aware that they work in a virtualized environment. One of the VMs takes the role of host and owns the GPU. Other VMs are guests and have access to the GPU through the host. Common methods are:

API remoting

In API remoting, each guest VM uses a proxy driver. The proxy driver forwards all user-space rendering or compute API calls to the host VM, where the real GPU driver executes the calls. This method is less efficient because each API call must go through a translation and transport layer.

Cooperative time slicing

In cooperative time slicing, each guest VM has a modified version of the real GPU driver. This modified driver shares access to the GPU with other VMs using time slicing. Each VM has direct access to the GPU registers and interrupts for a fixed amount of time. A central scheduling arbiter determines this time slice. The arbiter is commonly a software module running in the host VM.

Multi-GPU

In a multi-GPU system, each VM is given exclusive access to one of the GPUs on boot. Therefore there is no contention between VMs for access to GPU resources. The disadvantages are higher cost and limitation of the number of VMs to the number of GPU installed. Each VM is restricted to the GPU that is assigned on boot. Even if its compute

requirements become much higher than the other VMs, depending on workload, the VM cannot change GPU.

GPU partitioning

In GPU partitioning, a single GPU dynamically partitions its shader cores into isolated GPUs. This method has the same advantage as the multi-GPU method, but has the flexibility of dynamically adapting to the needs of each VM.

1.1 Mali GPU virtualization solutions

Arm provides several virtualization solutions for Mali™ GPUs.

Table 1-1: Virtualization solutions on Mali™ GPUs

Mali™ GPU	Paravirtualization	Hardware virtualization	Hardware separation
Mali™-G52	Yes	No	No
Mali™-G76	Yes	No	No
Mali™-G78AE	No	Yes	Yes

1.1.1 Paravirtualization with Mali™ GPUs

Arm provides a reference solution to support cooperative virtualization with Mali™-G52 and Mali™-G76 GPUs.

The GPU kernel driver in each *Virtual Machine* (VM) is aware of the virtualization and cooperates with an arbiter to share access to the GPU. A modified hypervisor performs the communication between kernel driver and arbiter, and also performs GPU switching between different VMs.

To support this design, the system must include:

- A virtualization supported GPU.
- A *System Memory Management Unit* (SMMU) or a *Memory Protection Unit* (MPU) inserted between the GPU and the system.



If memory isolation is needed between different VMs, an SMMU or MPU is required.

The system must also include software comprising:

- A hypervisor to route interrupts, register I/O, and support communication between the arbiter and VMs.

The GPU only has one set of interrupt request lines. The hypervisor must therefore track which VM is assigned on the hardware and route the interrupts accordingly.

The hypervisor uses the following procedure:

1. Reroute GPU interrupts to the active VM.
2. Remap GPU registers, allowing access from the address space of the active virtual machine.
3. Reconfigure the SMMU or the MPU.

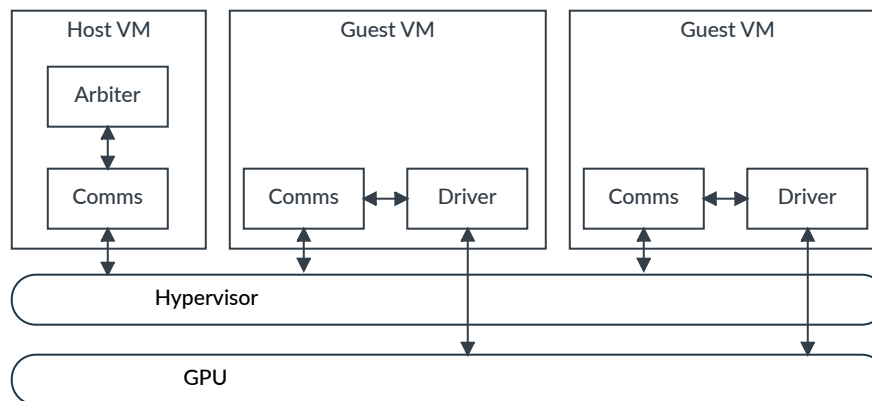


Out of security consideration, the hypervisor must manage the SMMU.

- System level software, including:
 - A bootloader
 - An operating system
- Virtualization components for the GPU, including:
 - The arbiter, a scheduling module that allocates the GPU to VMs.
 - An arbitration-enabled kernel driver for the GPU
 - Arbiter interface components
 - Arbiter message components

The diagram shows the outline of the system.

Figure 1-1: Software blocks in the paravirtualization system



1.1.2 Hardware virtualization

The Mali™-G78AE GPU includes a *partition manager* in hardware and extra system components required to support its operation.

The partition manager adds hardware support for access windows and hardware separation.

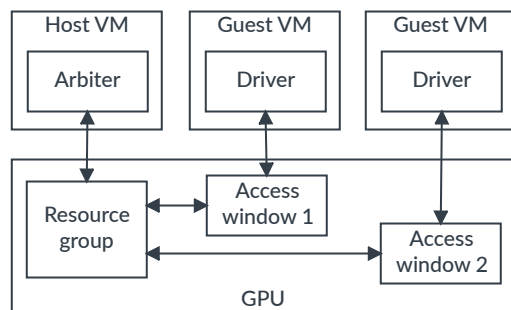
1.1.2.1 Access windows

An access window provides a *Virtual Machine* (VM) with access to a GPU. The access includes I/O registers, IRQ lines, and StreamIDs for the *System Memory Management Unit* (SMMU) or the *Memory Protection Unit* (MPU).

The arbiter controls multiple access windows and chooses which virtual machine, through its access window, has access to the GPU at any given time. The access window also provides a communication channel with the arbiter, which is always connected, so that the virtual machine can request access to the GPU.

The access window removes the requirement for the hypervisor to reroute GPU interrupts, remap GPU registers and reconfigure the SMMU or MPU when switching between VMs. Switching the current access window is enough. The access window also eliminates the need for the hypervisor to support communication between the VM and the arbiter. The figure shows the use of access windows in the GPU.

Figure 1-2: Access windows in the GPU



1.1.2.2 Hardware separation

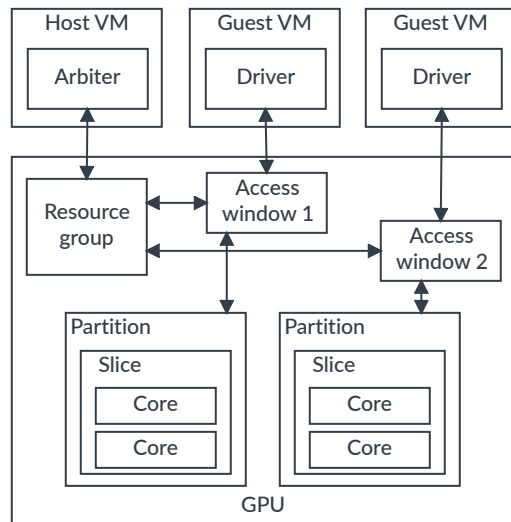
The partition manager makes it possible to separate a GPU into independent partitions, to effectively create multiple independent GPUs.

A Mali™ GPU with a partition manager groups shader cores into slices and groups slices into partitions. Assuming there are enough slices in the hardware, one or more slices can be assigned to a partition. Once configured, the partition can be treated like a conventional GPU.

Full hardware separation exists between the partitions, which means that different *Virtual Machines* can use the partitions independently. You can still use multiple access windows to share individual partitions cooperatively between multiple VMs.

Partitions and slices can be grouped into resource groups, which can in turn can be assigned to different bus interfaces.

Figure 1-3: Hardware separation in a GPU virtualization environment



1.2 Cooperative time slicing

For sharing a whole GPU or a GPU partition between multiple *Virtual Machines* (VMs), Arm implements a cooperative mechanism in the Mali™ kernel driver. This cooperation is marshaled by the arbiter.

The kernel driver in a VM requests access to the GPU from the arbiter when required. The driver must yield the GPU when the arbiter must assign it to another VM. For optimal performance, each

VM must yield the GPU or partition as quickly as possible when signaled by the arbiter. Rapid yielding minimizes GPU idle time and reduces latency for other VMs that have requested the GPU. To yield the GPU, the driver must:

- Suspend work on the GPU.
- Save any state required to resume the work the next time it gets GPU access.

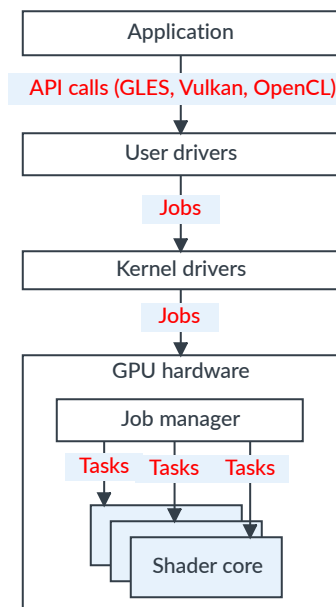
1.2.1 Mali DDK data-flow

API calls, issued by an application using the GPU, such as GLES, Vulkan, or OpenCL, are converted into one or more GPU jobs. There are different types of job, such as jobs for compute shaders, vertex shaders, fragment shaders, cache flushes, and others.

A job is converted to a collection of smaller tasks which are executed in the GPU. These tasks in turn are broken down into threads. A thread is an atomic unit of work executed by the shader core, and it corresponds to a single running instance of a computer shader, for example a vertex shader.

The following figure shows the simplified data-flow of the Mali GPU DDK. A job is completed when all corresponding tasks are completed. The *Job Manager* (JM), a hardware component on GPU, is responsible for scheduling jobs to run on the GPU.

Figure 1-4: GPU DDK data-flow



1.2.2 GPU yield process

In a system with GPU virtualization, the arbiter scheduler requests a *Virtual Machine* (VM) to yield the GPU when its time slice has expired and another VM requires it.

The whole GPU yield process is:

- The arbiter sends an interrupt to the VM signaling it to yield.
- The VM kernel driver:
 - Processes the interrupt.
 - Stops submitting new jobs to the GPU *Job Manager* (JM).
 - Writes `SOFT_STOP` to the `JOB_SLOT_CONTROL.COMMAND` register.
- The job manager on the GPU must:
 - Stop submitting new tasks to the shader cores.
 - Wait for the shader cores to complete their pending tasks.
 - Flush the L2 cache to DRAM, along with any internal state required to resume the jobs.
- The VM kernel driver sends an interrupt to the arbiter to signal it has yielded.

Potential bottlenecks:

- The execution time of a pending GPU task. Individual tasks cannot be interrupted, and so the job manager must wait for them to complete. This restriction means that the kind of jobs that applications generate can adversely affect the time for a VM to yield the GPU to the arbiter. For example, a very expensive fragment shader with long running loops accessing a lot of memory can significantly affect the execution time of a single task.
- Memory bandwidth. The L2 cache has to be written to DRAM when yielding, which can be multiple tiles of memory. This issue includes waiting for the transfer to complete. Therefore the system memory bandwidth is a factor in how expensive a yield is. This issue is also true when a job resumes when the VM next gets GPU access. The resumption involves reading the content in DRAM back into the L2 cache.
- Interrupt latency. Only a single interrupt can be handled at a time. Therefore, the number of other interrupts generated throughout the rest of the system can adversely affect the yield time. This issue is not as significant when using a real-time OS or the `PREEMPT_RT` patches for Linux, which allow interrupt handling to be done asynchronously.

GPU lost

The arbiter only waits a finite amount of time for the VM to yield the GPU. When this time expires, the arbiter forcibly removes access to the GPU from that VM, and sends a `GPU_LOST` message to the VM. This leaves the jobs in an unknown state, and the jobs cannot resume. The VM kernel driver must then communicate this to the user-space application.

1.2.3 API usage and yielding

Two factors impact the execution of an application using the GPU and running in a paravirtualized solution.

- The length of the time slice that is allocated to a *Virtual Machine* (VM) to run a graphic render pass or a compute task. To avoid reducing the frame rate, this length of time is typically between 4ms and 8ms for graphic applications.
- Developer-specific optimization in the API invocation, to make the render pass or the compute task fit in the allocated time slice.

Graphics shader complexity

The shader complexity and execution times directly influence soft-stop time. Once the GPU starts the shader execution, the execution cannot be suspended.

The driver does not have any means of knowing how a shader can be efficiently broken down into smaller pieces. To ensure that the GPU can quickly perform soft-stop, the shader complexity must be appropriate. You can use the off-line shader compiler to assess the complexity of shaders.

Compute shaders complexity

With compute shaders, the developer must ensure that the shader complexity is appropriate for the time slices allocated to the VM.

Tasks size

Draw-call size can impact the size of vertex jobs.

- It is important to avoid putting too many vertices in a single draw-call. Breaking vertices into smaller draw-calls is a better solution.
- Currently, for the Job Manager based GPUs, the hardware only supports soft-stop of *Index Driven Vertex Shading* (IDVS) jobs at job boundaries.
- The driver cannot optimize indirect draw-calls, because the driver does not know the size of such draw-calls at submission time.

For fragment shader jobs, a task is a full fragment tile which cannot be suspended mid-execution. The full tile must complete before the job can be suspended. The workload in the tile therefore defines the suspend time.

Jobs are split into tasks before being effectively executed in parallel by the GPU. The number of tasks in the shader queue is one of the important factors that affect the soft-stop time. It is therefore important to choose the right task split when the API allows this operation. Indeed, smaller tasks sizes mean shorter soft-stop time. However, more tasks in the shader queue can also increase the soft-stop time. You must find the right balance for task size depending on the application characteristics and targeted performance.

Compute tasks are suspended at task granularity. A task comprises one or more workgroups. `cl_arm_scheduling_controls`, an OpenCL extension, allows the number of workgroups per task to be defined from the API level. With this extension, you can set the batch size of workgroups. The

smaller the batch size is, the shorter the soft-stop time it requires. Again, a balance must be found because a small batch size, providing a short soft-stop time, results in low utilization of the GPU.

**Note**

Avoid the situation where a sequence of API calls can never be executed in the time slice allocated for a VM. If a task requires more time than the time slice allocated for the VM, the execution can never complete.

1.2.4 Tiler queue size

The tiler queue size can affect the yield times by affecting the soft-stop times. A soft-stop completes when all the jobs in the queue are finished. If there are longer queues, it tends to increase the time spent in the soft-stop process. Longer queues are more likely to affect soft-stop times when the GPU processes geometry-heavy contents.

For GPUs with hardware that allows 8 queued jobs in pipelined queues, it can potentially take longer time on soft-stop, than other GPUs that allow a maximum of 4 jobs in the pipeline. For this reason, the reference DDK code has a build option called `LIMIT_TILER_JOB` to limit the tiler queue size to 4. With this option, you can guarantee that in a worst case, the number of jobs during a soft-stop is equivalent to other GPUs that allow a maximum of 4 jobs in the queue. In other words, it gives flexibility trying to keep the soft-stop times at a desirable level for geometry-heavy contents, when the number of pipelined jobs considerably affects the yield times.

**Note**

Mali™-G78AE supports `LIMIT_TILER_JOB`.

For more details about `LIMIT_TILER_JOB`, see the *Driver build options* section in the *Parameters to tune DDK performance* appendix in the *Arm® GPU DDK Integration Manual*.

When using `LIMIT_TILER_JOB`, consider the following points:

- It aims at improving yield times and improving the overall performance of virtualized environments where the GPU is time sliced between multiple VMs. So, it must not be used in non-virtualized setups where yield times are not relevant.

**Note**

For a non-virtualized GPU, do not use this option.

- Although this optimization is likely to have a positive impact in decreasing the yield times for geometry-heavy contents, it may affect the processing performance of the workload on the GPU, once the number of queued jobs is limited, depending on many factors, like contents the GPU processes or the number of share cores available.

- For using this optimization, you must evaluate the content and compare the overall performance before and after enabling `LIMIT_TILER_JOB` to ensure the performance improves.

1.2.5 Choosing the time slice length

Parameters and constraints implied by the application are important for length of time slice.

Consider these factors when choosing the time slice length:

- The target rendering frames per second (FPS) for a graphic application.
- The number of *Virtual Machines* (VMs) using the GPU in the system. Typically, if the system is composed of N virtual machines with each rendering at 60 FPS, the time slice is at approximately $16\text{ms}/N$.
- The maximum time for running protected content jobs.



Where applications must not cause a GPU_LOST message, you must consider the worst case soft-stop time when configuring the time slice length.

Where applications are able to handle a GPU_LOST interruption, the time slice length can be set based on the average soft-stop times.

2. Reference software stacks for virtualization

A reference software for paravirtualization includes various kernel modules, including an arbiter.

The reference stack for Mali™-G78AE is provided as example code under GPLv2 and a permissive MIT license, and can be ported as necessary.

The reference example permits the Device tree to contain the hardware definition, with clear interfaces between modules.

2.1 Overview

Four main functionalities are required to enable the virtualization:

Arbiter

A software module to coordinate access to the GPU hardware.

Cooperative driver

A standard GPU kernel driver with an interface to communicate and coordinate GPU access with the arbiter.

Communication

A system to allow two-way communication between the arbiter and the GPU kernel driver instances.

Power management

A system to control GPU power management that accounts for the compute load of all the GPU kernel driver instances.

The following sections give an overview of each of these functionalities and how they appear within the reference implementation.

2.1.1 Arbiter

The arbiter maintains a list of all the GPU kernel driver instances.

The arbiter grants the GPU for specific time interval to each GPU kernel driver instance and handles GPU kernel driver instances that do not release the GPU within a given time.



The actual behavior of arbiter scheduling is not mandated. The behavior must be implemented to suit the use cases of any given system.

The reference implementation of an arbiter is a kernel module, called `mali_arbiter`. This module does not require direct access to the GPU hardware. Instead, this module is provided with interfaces to delegate hardware and platform interactions to other modules. The interactions include communicating with GPU kernel driver instances, or switching the GPU to different GPU kernel driver instances.

For clarity, the reference implementation of the arbiter just schedules each GPU kernel driver instance with an equal time-slice. This type of scheduling is also known as mediated pass-through or time-slice scheduling. The following considerations might influence the changes in the reference arbiter scheduling policy.

- Specific use-case knowledge, such as priority of a task and minimum yield time.
- The number of *Virtual Machines* (VMs) in the system.
- Knowledge of rendering time.
- The latency of the arbiter, and communication with the arbiter, can directly impact the system performance.
- The arbiter controls the GPU and can affect the *Quality of Service* (QoS) of the *Virtual Machines* (VMs). The arbiter can potentially cause a *Denial of Service* (DoS) through poor scheduling.

2.1.2 Cooperative driver

GPU driver cooperation is only required on the kernel side, so the user side GPU driver does not need any modifications.

The GPU kernel driver must request access to the GPU and wait for the arbiter to grant the access. The kernel driver must also release this access when required, so the arbiter can grant it to another GPU kernel driver instance.

The reference implementation of the GPU kernel driver is a kernel module called `mali_kbase`. This module has been modified to implement these functions:

- `mali_kbase` can request the GPU from the arbiter when the arbiter has work to do, and release the GPU when the arbiter requests the access back.
- When releasing the GPU, `mali_kbase` has to stop submitting jobs to it and wait for the jobs currently executing to complete, which is called a soft-stop.
- If the GPU kernel driver fails to release the GPU in the allotted time, the arbiter can hard-reset the GPU, erasing the pending and executing jobs. The GPU kernel driver signals an error. This error is reported back to the application through the user-space API.

2.1.3 Communication

For cooperative virtualization, a communication channel is required between the arbiter and the GPU kernel driver instances inside the *Virtual Machines* (VMs).

This communication is through well-defined interfaces. For the reference implementation, the arbiter and supporting functions are implemented as kernel modules. The interface is added to the user-data slot of each driver as `drvdata`. `drvdata` is the Linux kernel construct that holds the interfaces and is accessed through `dev_get_drvdata()`. Modules can only communicate with each other if they are in the same VM. Any module can call these structures by fetching the `drvdata` for the required device. If you require communication between modules in different VMs, then you must implement a suitable alternative to these function pointers.

For the reference implementation, the GPU kernel driver instance uses an interface to talk to the arbiter module called `arbiter_if_vm_arb_ops`. The primary functions provided by this interface are:

- Register VM with the arbiter, and provide the arbiter with an interface it can use to talk back to this GPU kernel driver instance.
- Request ownership of the GPU.
- Release ownership of the GPU.

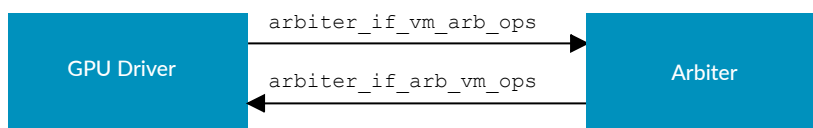
To allow communication from the arbiter to the VM, the GPU kernel driver instance implements an interface called `arbiter_if_arb_vm_ops`. This interface is passed to the arbiter when registering the GPU kernel driver instance. The primary functions provided by this interface are:

- Grant ownership of the GPU to the GPU kernel driver instance.
- Request that the GPU kernel driver instance release ownership of the GPU.

Connection of a GPU driver instance to the arbiter is done through intermediate kernel drivers. This connection is defined in the `gpu` node of the kernel Device tree. The phandle `arbiter_if` links the GPU kernel driver instance to a communication driver that implements the `arbiter_if_vm_arb_ops` interface.

For example, the figure shows how the interfaces can be called in a simple arrangement. In this arrangement, there is only one GPU kernel driver instance, and it is in the same VM as the arbiter.

Figure 2-1: Driver to arbiter simple arrangement



However, virtualization is intended to share the GPU hardware with multiple GPU kernel driver instances on multiple VMs. Therefore, a communication bus or shared memory is required to enable communication with multiple VMs. The figure shows a complete driver to arbiter arrangement.

Figure 2-2: Driver to arbiter complete arrangement

The messaging protocol attached to the GPU kernel driver implements the `arbiter_if_vm_arb_ops` interface, implementing the interface to the arbiter. While the messaging protocol attached to the arbiter implements the `mali_vm_ops` interface. The messaging protocol translates the interface operations between function calls and message packets, similar to the way remote procedure calls work.

The implementation of the communication between the arbiter and the GPU kernel driver instances depends on:

- Which hypervisor is used.
- Whether a GPU with hardware virtualization is used.

2.1.4 Switching the GPU

When the arbiter is ready to pass ownership of the GPU to a particular GPU kernel driver instance, the arbiter uses an implementation-specific interface.

The implementation of this interface depends on which hypervisor is used, and also depends on whether a GPU with hardware virtualization is used.

2.1.5 Power management

In a non-virtualized environment, there is only one GPU kernel driver instance using the GPU. The GPU kernel driver feeds its activity directly to the power management interface of the platform.

The power management subsystem uses this information to:

- Adjust the GPU clock frequency.
- Perform *Dynamic Voltage and Frequency Scaling* (DVFS).
- Turn the entire GPU on and off.



Note

The entire GPU power control is only available in a paravirtualized environment and hardware virtualization with a single OS configuration. For hardware virtualization on multiple OS configurations, the partition manager must remain powered on to retain its state and allow communication between the arbiter and the access windows.

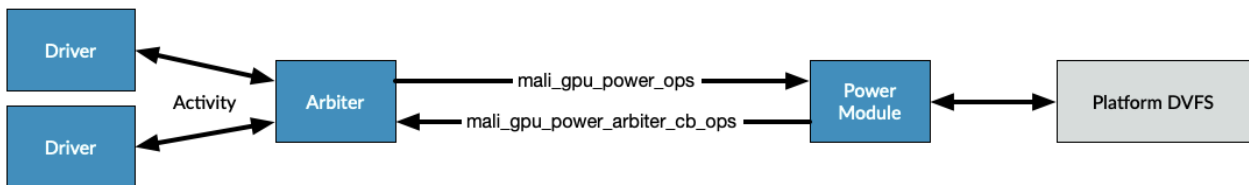
In a virtualized environment, each GPU kernel driver instance only knows about its own activity. The driver has no knowledge of the power requirements of the other GPU kernel driver instances. For this reason, the power management features in the GPU kernel driver are disabled.

Instead, this responsibility is delegated to the arbiter. The arbiter in turn delegates power requirements to another module called `mali_gpu_power`. This module feeds the total activity of all the GPU kernel driver instances to the power management interface of the platform. This module:

- Abstracts this platform-specific functionality away from the arbiter logic.
- Covers the scenario where there might be multiple arbiters in a system sharing different parts of the same GPU. In that scenario all the arbiters would connect to the same power module.

The logic is represented in this figure.

Figure 2-3: Arbiter and power module



In the figure, the GPU kernel driver to arbiter communication layer has been omitted for clarity. The arbiter talks to the power module using the `mali_gpu_power_ops` interface, while the power module talks back to the arbiter using the `mali_gpu_power_arbiter_cb_ops` interface.

The power module in the reference implementation does the following:

For GPU power control

When the system boots up, the power module requests the powerup of the GPU. After this request, the hardware power management begins. When none of the arbiters report to the power module that their kernel driver instances are using the GPU, the module can power the unit down. This action saves power.

For Dynamic Voltage and Frequency Scaling

Depending on the usage of the GPU, DVFS can change the frequency of the unit to meet computational demands. In the virtualized system, the power module computes the time that the GPU kernel driver instances connected to each arbiter take using the GPU. Each arbiter tracks how long all or part of the GPU has been scheduled to a GPU kernel driver instance. The power module requests this utilization data from each arbiter when the kernel requests it. The power module sets the GPU frequency accordingly. Specific use case requirements can help optimize the scaling calculation.

2.2 Paravirtualization

Paravirtualization uses software to virtualize the GPU hardware, assisted by a modified hypervisor.

When you use paravirtualization, a hypervisor is required to perform the following operations:

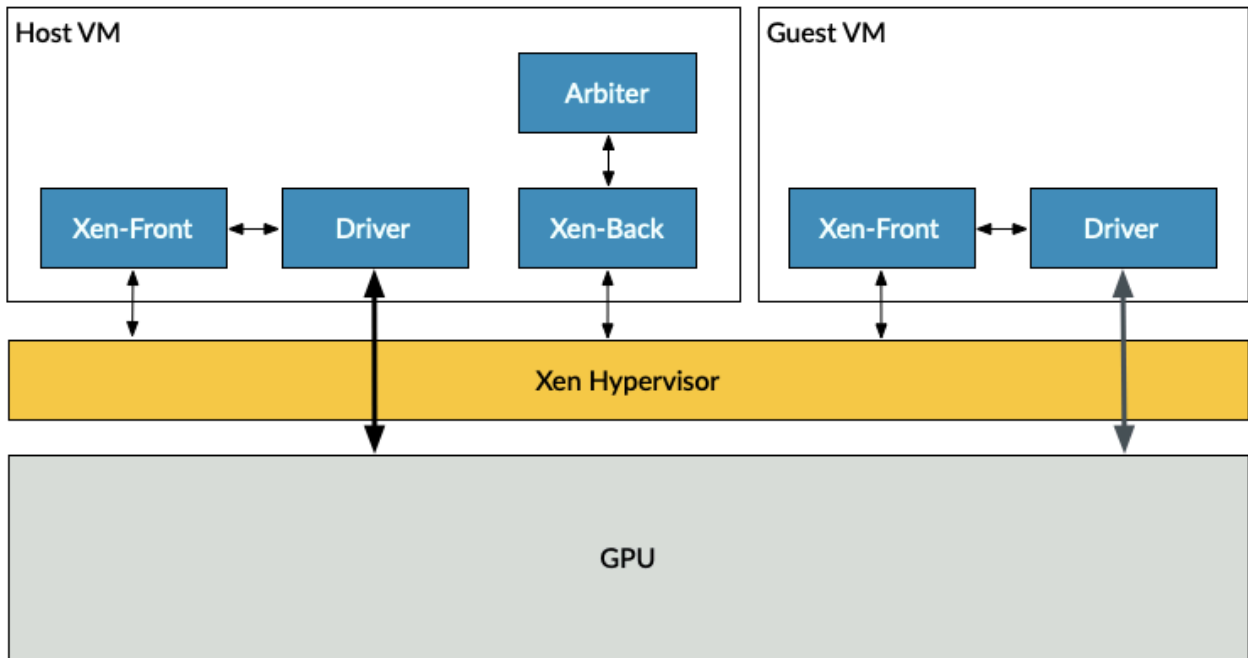
- ◦ Reroute the GPU interrupts to a VM.
- ◦ Remap I/O registers into a VM address space.

- Optionally control an SMMU for GPU memory access.
- Sending messages between VMs and the arbiter, by using the XenBus combined with a front-end and back-end split Xen kernel driver.

For more details, see [B. Xen hypervisor](#) on page 102.

The figure shows a system with two VMs, a host VM and a guest VM. The figure shows the communication in this scenario.

Figure 2-4: Host VM and guest VM



The figure shows the arbiter and GPU kernel driver instance in the same VM. This system is only an example and not a requirement.

This figure shows the communication interfaces when the paravirtualization reference implementation is used. For more details about other options, see [2.6 Example use-cases](#) on page 30.

Figure 2-5: Paravirtualization for driver and arbiter



2.2.1 Paravirtualization time slice metrics

The Mali™ Linux kernel driver (`kbase`) generates kernel events as trace-points from the host OS and the *Virtual Machine* (VM) OS to help analyze the VM time slice on paravirtualization stack.

To obtain the kernel events, use `kbase` built using the `MALI_DEBUG=y` file and the Linux kernel ftrace instrumentation.

The following table describes qualifiers for kernel events.

Table 2-1: Kernel events

Event qualifier in nanoseconds (ns)	Description
<code>LAST_STOP = (last ARB_GPU_STOPPED or 0)</code>	Timestamp of the last stop.
<code>ARB_GPU_GRANTED</code>	Timestamp of the next <code>ARB_VM_GPU_GRANTED</code> message.
<code>ARB_GPU_STARTED</code>	Timestamp of the next <code>KBASE_VM_STATE_STARTING</code> or <code>KBASE_VM_STATE_STOPPING_ACTIVE</code> state.
<code>ARB_GPU_STOP_REQUESTED</code>	Timestamp of the next <code>ARB_VM_GPU_STOP</code> message.
<code>ARB_GPU_STOPPED</code>	Timestamp of the next <code>KBASE_VM_STATE_STOPPED</code> state.
<code>ARB_GPU_LOST</code>	Timestamp of the next <code>ARB_VM_GPU_LOST</code> message.

The following table describes qualifiers for time ranges and formulas.

Table 2-2: Time ranges and formulas

Time qualifier in microseconds (μs)	Description
<code>WAIT_TIME = (ARB_GPU_GRANTED - LAST_STOP)</code>	Time between <code>LAST_STOP</code> and <code>ARB_GPU_GRANTED</code> . Time waited by <code>kbase</code> for the GPU. First entry is invalid.
<code>START_TIME = (ARB_GPU_STARTED - ARB_GPU_GRANTED)</code>	Time between <code>ARB_GPU_GRANTED</code> and <code>ARB_GPU_STARTED</code> . Time required by the driver to start a job.
<code>RUN_TIME = (ARB_GPU_STOP_REQUESTED - ARB_GPU_STARTED)</code>	Time between <code>ARB_GPU_STARTED</code> and <code>ARB_GPU_STOP_REQUESTED</code> . Time spent executing on the GPU.
<code>STOP_TIME = (ARB_GPU_STOPPED - ARB_GPU_STOP_REQUESTED)</code>	Time between <code>ARB_GPU_STOP_REQUESTED</code> and <code>ARB_GPU_STOPPED</code> . Time required by the driver to stop a job.
<code>TOTAL_SWITCH_TIME = (START_TIME + STOP_TIME)</code>	Total switching time (that is, Start Time + Stop Time). This time is also known as yield time (<code>YIELD_TIME</code>).

2.3 Hardware virtualization

Relying on a hypervisor to perform GPU resource management and communication is computationally expensive. You can use a GPU with hardware virtualization to improve the efficiency.

The Mali™-G78AE GPU has a hardware component called the partition manager which accelerates the following operations:

- Switching the GPU interrupts, registers, and address space between different GPU kernel driver instances with a fixed hardware block, called the access window.
- Sending messages between GPU kernel driver instances and the arbiter with message registers in the access windows.

To use this acceleration, the GPU exposes the following hardware blocks:

Access window

The access window maps a set of GPU registers, interrupts, and StreamIDs to the GPU kernel driver instance. Only StreamIDs are used to tag the stage 2 address lookups in the SMMU.

Provides an incoming and outgoing message register for communication with the arbiter, or more specifically with the resource group.

Resource group

The resource group is one of the hardware backends for the arbiter, used to communicate with the GPU kernel driver instances through their access windows.

Provides an incoming and outgoing message register for communication with up to 16 access windows.

There is one resource group for each arbiter.

Configuration and partition control

The partition configuration and partition control blocks are the hardware backends for the arbiter. The blocks switch all or part of the GPU to a particular access window. For more details about partitioning, see [2.4 Hardware separation](#) on page 24.

There can be one or more sets of partition control instances per resource group.

System

The system block configures StreamIDs used to tag memory transactions to enable an SMMU to provide memory isolation memory protection for VMs. There are 2 StreamIDs for each of the access windows, one for protected transactions and one for normal transactions.

Reports system faults.

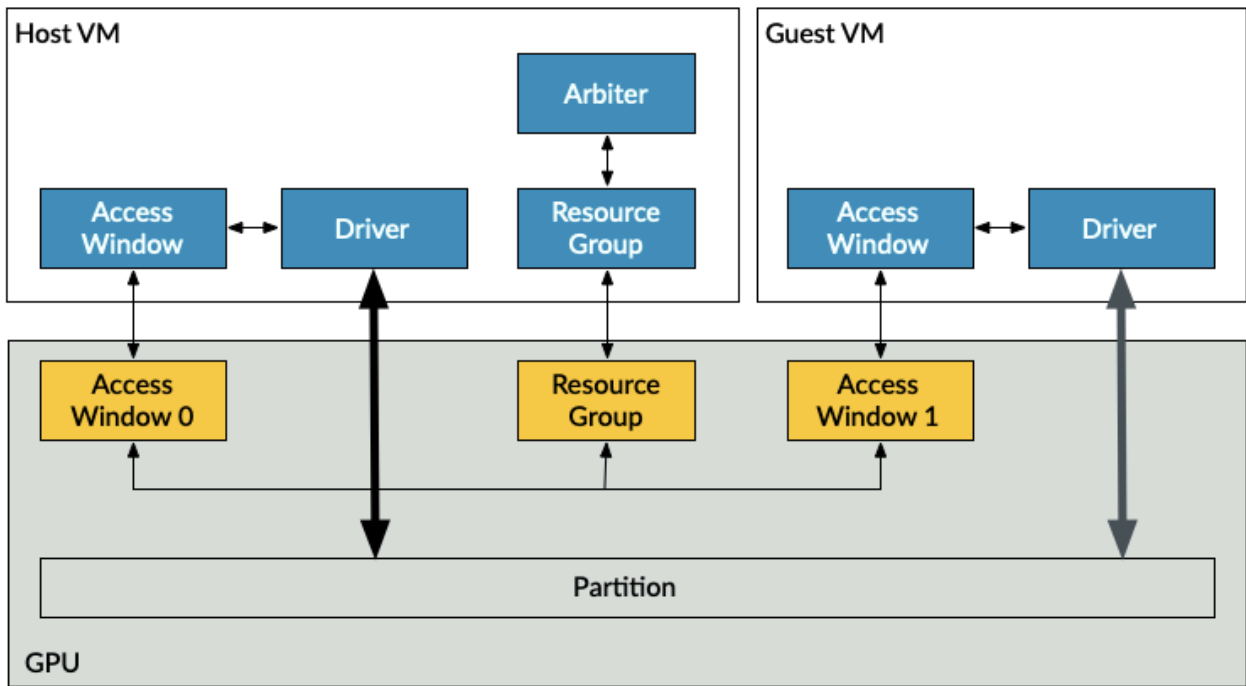
Power management

The reference stack supports the suspend to RAM power state in a single OS domain with the device link framework of the Linux kernel. The reference stack does not support the suspend to RAM power state for multiple OS domains.

Each of the hardware blocks is aligned to their own memory page. This alignment permits the blocks to be moved to different VMs as required in your system design. In the reference implementation, a separate kernel module is used to manage each hardware block.

The figure shows a scenario involving two VMs.

Figure 2-6: VMs and partition



The figure shows the communication in a scenario with hardware virtualization for driver and arbiter.

Figure 2-7: Hardware virtualization for driver and arbiter



2.4 Hardware separation

Paravirtualization and hardware virtualization are ways of cooperatively sharing a single GPU shared between multiple GPU kernel driver instances.

Hardware separation permits you to divide available GPU resources into partitions that are prevented from interfering with each other.

This permits safety or timing critical workloads to run without the indeterministic impact of sharing GPU resource with other VMs.

Hardware separation (partitioning) and time slicing are not mutually exclusive. For example, you can choose to have one partition that is shared with multiple VMs and one that is dedicated to one VM.

Hardware separation on the GPU

The Mali™-G78AE has a partition manager hardware that permits the GPU to be partitioned into several independent GPUs. Each of these GPUs has its own tiler and shader cores. The number of the GPU shader cores allocated to each of these partitions is dynamic. You can change the number of cores allocated to a partition at runtime with arbiter reconfiguration.

The advantages of this approach are:

- Safety critical VMs can have completely isolated and fixed GPU resources.
- More cost effective and power efficient than multiple GPUs.
- GPU cores can be dynamically allocated to VMs based on usage, by switching the access window to a larger partition, or resizing its current partition.

To implement hardware separation, a GPU with a partition manager organizes its resources in the following ways:

Slice

Shader cores are grouped into hardware units. The units are called slices.

Each slice has its own tiler.

Slices have a fixed number of shader cores, and the hardware implementation sets this number.

The atomic unit for allocating GPU resource is the slice.

Partition

A partition can be considered as an independent GPU.

One or more slices can be dynamically grouped into partitions.

When multiple slices are assigned to a single partition, only the tiler from the first slice is used. The tilers belonging to the other slices in the partition are disabled.

AXI buses

The GPU is accessed with one of 3 AXI busses, AXI-A, AXI-B, and AXI-C. The bus separation is provided to support the safety critical use cases.

AXI-A

Typically, AXI-A is used by a non-critical cluster. It runs quality-managed software, for example quality-managed VMs, such as Android.

AXI-B

Typically, AXI-B is used by a safety critical cluster. It runs safety critical workloads.

AXI-C

Typically, AXI-C is used by a safety island. It is responsible for allocating resources and handling errors.

To take advantage of this hardware separation, some extra hardware blocks are exposed:

Assign

One for the entire GPU.

Statically assigns resource groups to AXI buses.

Statically assigns slices, partitions, and access windows to individual resource groups.

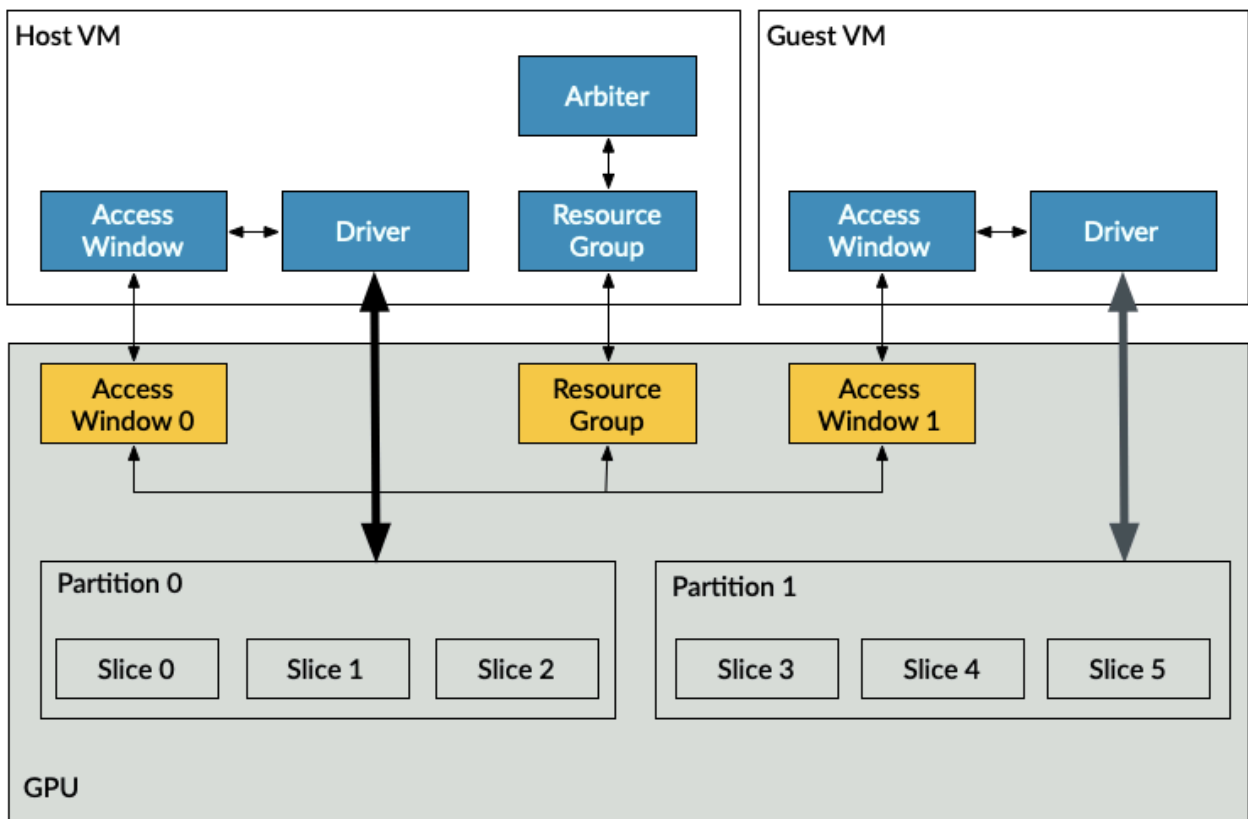
Configuration and control

One or more for each resource group.

For a safety critical system, the safety island is expected to configure the system and assign pages before the VMs boot.

The figure shows the scenario involving two VMs with each having its own partition.

Figure 2-8: VMs partition for hardware separation



2.4.1 Partition management

The reference arbiter provides a `sysfs` interface to manage partitions and access windows.

In a production system, you must use an implementation-specific mechanism for using partition. Considerations when designing your implementation are:

Safety requirements

Each safety critical *Virtual Machine* (VM) must be able to assign to its own partition without allowing that partition to be time-sliced with any other VMs.

Performance requirements

The activity of a particular VM can be analyzed at runtime by measuring how long it takes to release the GPU. If this VM does not appear to be able to cooperate well with other VMs, its access window can be moved to its own partition. Or, if a partition is not delivering enough performance, the number of slices allocated to it can be increased dynamically at runtime.

2.5 Resource assignment with the partition manager

The GPU uses the partition manager to allocate graphics processing resources in groups for the physical partitions.

In a typical system, a hardware manager, running on the safety island, specifies which resources each group can use. The steps to assign the GPU resources through the assignment registers are:

1. Assign each group to a bus according to which application processor accesses it (PTM_RESOURCE_GROUP_BUS).
2. Assign each partition to its group (PTM_PARTITION_RESOURCE_GROUP).
3. Assign each slice to its group using PTM_SLICE_RESOURCE_GROUP. An arbiter can assign any slice within its group to any partition within the same group.
4. Assign each access window to its group with PTM_AW_RESOURCE_GROUP. An arbiter can assign any access window within its group to any partition within the same group.
5. Assign each access window a protected and a not-protected StreamID. The StreamID tags each DRAM transaction (PTM_AW0_STREAM_ID and PTM_AW0_PROTECTED_STREAM_ID).

PTM_RESOURCE_GROUP and PTM_PARTITION_CONFIG registers. The steps to do so are:

1. Assert reset to the GPU processing slices. The slices are reconfigured through PTM_SLICE_RESET_SET and PTM_SLICE_RESET_STATE.
2. Disable clocks to the GPU processing slices being configured. The slices are reconfigured through PTM_SLICE_CLOCK_SET and PTM_SLICE_CLOCK_STATE.
3. Update PTM_SLICE_MODE_NEW.
4. Assert PTM_SLICE_MODE_UPDATE.
5. Poll PTM_SLICE_MODE_ACK to confirm the change has completed.
6. Deassert reset to GPU slices. The slices are reconfigured through PTM_SLICE_RESET_SET and PTM_SLICE_RESET_STATE.
7. Enable clocks to the GPU processing slices being configured. The slices are reconfigured through PTM_SLICE_CLOCK_SET and PTM_SLICE_CLOCK_STATE.

The slices not only dedicated to graphics are assigned to the same group as the partition can be enabled for that partition. Read-only registers in each group region show which slices, partitions,

and windows can be used. If the number of slices in a partition changes, the GPU must inform the driver. The driver must then adjust memory allocations that depend on the number of shader cores. For example, the driver must change thread-local storage. Alternatively the driver can allocate resources assuming each partition has the maximum configuration of shader cores.

When a graphics processing slice is configured as a secondary slice it exchanges data with the preceding slice. When configured as a primary slice it ignores the preceding slice. If an arbiter incorrectly configures a primary slice to be a secondary slice, the slice might interfere with the neighboring partition. Therefore the PTM_ASSIGN.PTM_SLICE_ISOLATION register, controlled by the safety island, enforces isolation between partitions in different groups.

Each instance of the GPU driver can use a private address window through which it accesses a partition. It is possible for separate driver instances to share a window, though not at the same time, when system software provides support. The PTM_AW_SET register, in the partition control group, determines which window is active for a partition. This register determines to which partition a transaction is sent, for a transaction received through that window. It is illegal to enable the same window in more than one partition or to enable more than one window for a given partition at the same time. If this condition is not met, the unit reports an error reported through PTM_PARTITION_STATE.

The steps to change which window has access to a partition use the corresponding PTM_PARTITION_CONTROL registers:

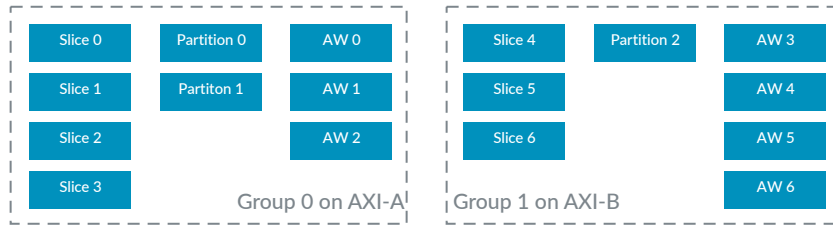
- When the partition is in use, software must request the owning driver instance to release it (yield process).
- Wait for the driver owning the instance to yield the partition through PTM_RESET_SET.
- Wait for PTM_RESET_STATE to indicate that the reset has been applied.
- Enable an access window by setting PTM_AW_SET.
- Grant the driver requesting instance access to the GPU (software handshake).

Only windows that have been assigned to the same group as a partition can be enabled for it. The partition reset prevents information leaking from one driver instance to another. A lock that write-protects all partition configuration and most control registers, enforces the reset. Only PTM_RESET_SET can be modified while a window is enabled. Such modification releases the lock and clears the PTM_AW_SET register to ensure a virtual machine cannot continue to access the partition when it is reset. The state of the lock is in the PTM_STATE register. The purpose of the lock is to remove the need to prove the arbiter cannot interfere with an active partition other than by resetting it. If a driver accesses a reset partition it receives an error interrupt, alerting it to the action. Because the partition clock and power control registers cannot be modified while a window is active, they are not suitable for driver power management. Power management is achieved through clock and power-gating registers in the partition job manager.

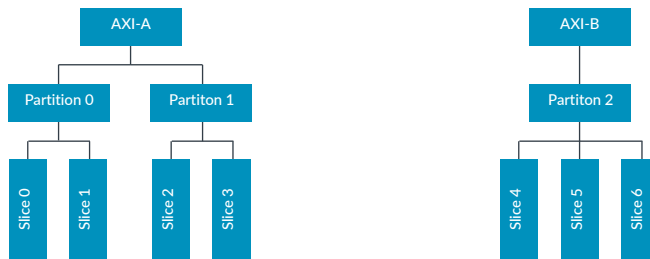
The figure shows the steps to assign and control resources.

Figure 2-9: Assignment of resources with the partition manager

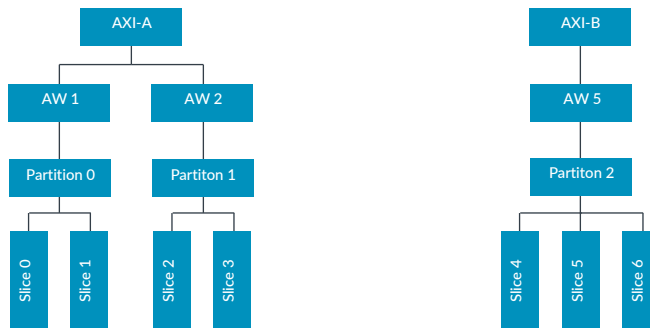
Step 1: assign each group to a bus, then add slices, partitions, and access windows to each group.



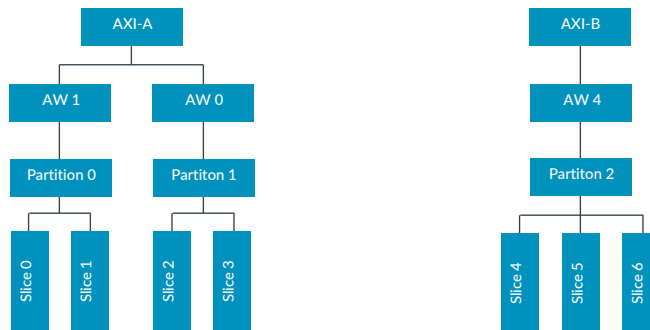
Step 2: assign slices into partitions to make GPUs.



Step 3: enable an access window for each partition to allow a driver access to it.



Step 4: allow different drivers to take turns on partitions by changing the access windows.



2.5.1 Arbiter messaging in the partition manager

The GPU driver communicates with the arbiter to gain access to the GPU. The details of the communication API change for different arbiters.

The partition manager standardizes the Communications channel through hardware message registers. Each access window has two 64-bit registers, one for sending messages to the arbiter and one for receiving them. Each partition has a corresponding set of registers for each access window. An interrupt is raised when a message is received.

Control registers:

- Enable interrupts.
- Permit software to inspect and clear interrupts.
- Permit software to inspect the status of outgoing messages. When the receiver clears its interrupt, it indicates receipt of the message.

Message interrupts are level-sensitive. If a message is received when an interrupt is disabled, it is signaled when the interrupt is next enabled. Only the message registers, belonging to an access window and assigned to a group, can be read or modified by the arbiter for that group. Other attempted writes are discarded, and reads return zero. The message registers for an access window remain accessible and active when the window is disabled. The driver specifies the messages and the protocol for exchanging windows.

2.6 Example use-cases

Various arrangements are possible for hardware virtualization with GPUs with both single and multiple CPUs.

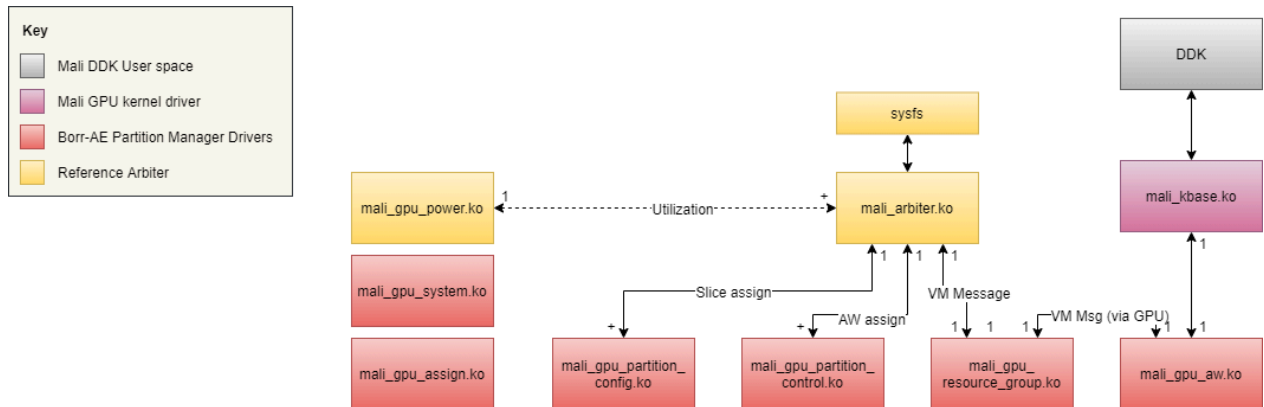
The reference stack has been designed with flexibility in mind. Each of the modules perform tasks specific to a hardware block. This separation permits modules to be moved to different VMs, OSs, or domains as required by the customer. As such, there is no set way for the system to be configured. This section provides some examples of how the reference stack might be changed to suit a given application.

Use case examples of virtualization for the Mali™-G78AE GPU

Example use cases for the Mali™-G78AE GPU with hardware virtualization.

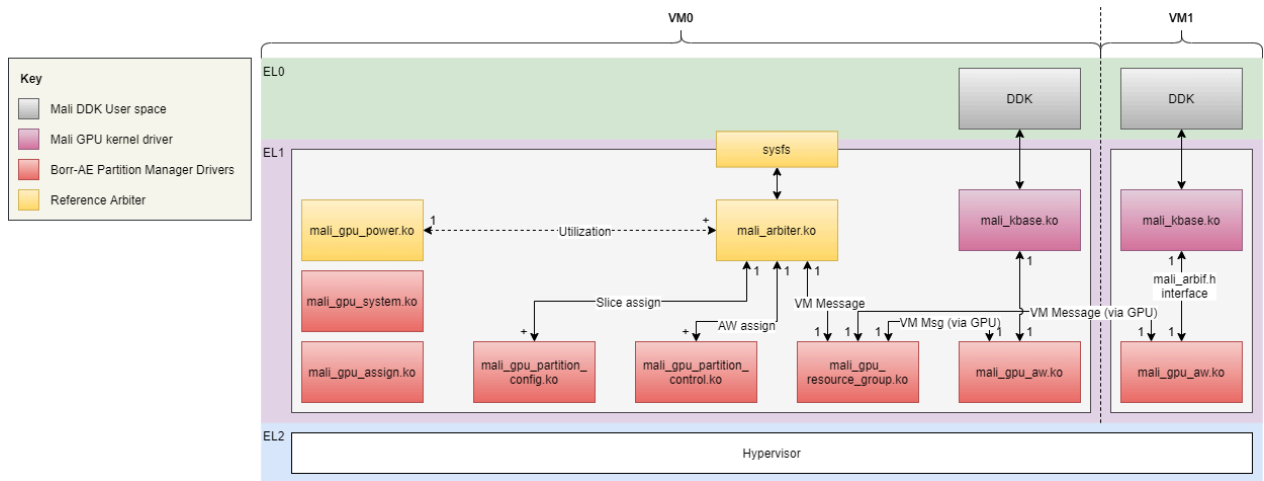
Single CPU cluster configuration with single OS

The figure shows the most basic configuration that involves using Mali™-G78AE GPU inside a single OS. The arbiter and several support modules are required alongside `mali_kbase.ko` to set up the device.

Figure 2-10: Single CPU cluster configuration with single OS for Mali™-G78AE GPU

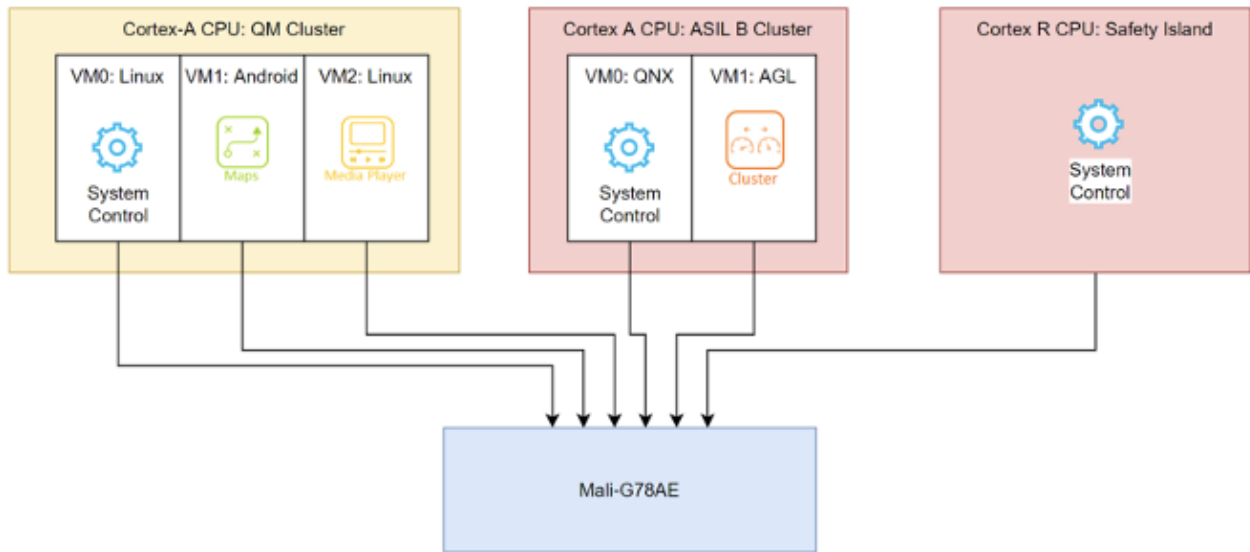
Single CPU cluster configuration with multiple VMs

The figure shows that a VM is required for the arbiter and support modules. Other VMs can load the mali_gpu_aw and mali_kbase modules to access a partition of the GPU. The VM containing the arbiter can also load these modules to use the GPU if necessary.

Figure 2-11: Single CPU cluster configuration with multiple VMs for Mali™-G78AE GPU

Multiple CPU clusters

In the following example configuration, the QM, and ASIL clusters access the Mali™-G78AE GPU through the AXI-A and AXI-B buses. A safety island is used to manage how the GPU slices are partitioned.

Figure 2-12: QM and ASIL clusters access Mali™-G78AE GPU

The following example shows a software configuration with multiple CPU clusters.

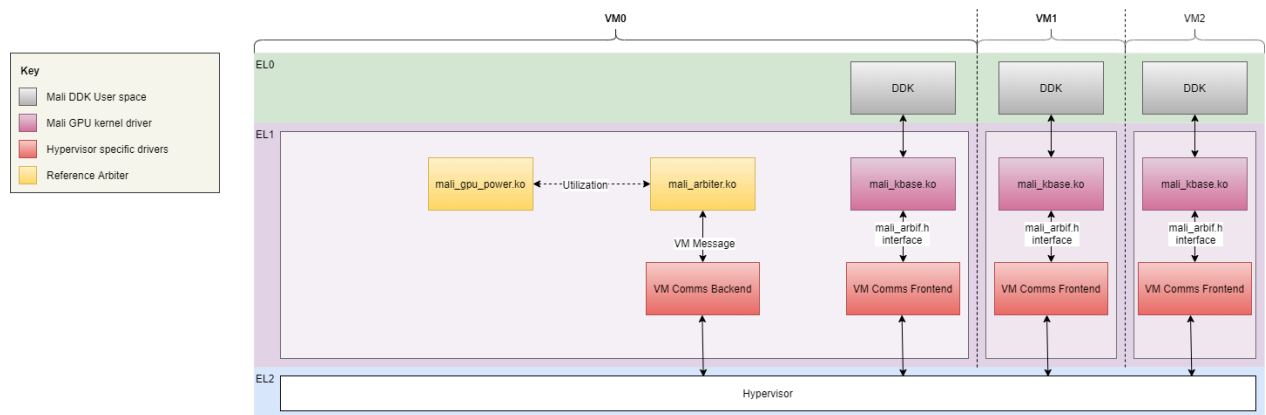
Figure 2-13: Software configuration with multiple CPU clusters

Use case examples for paravirtualization

For GPUs that do not support hardware virtualization, software paravirtualization can be used instead. Here are some examples of possible software configurations.

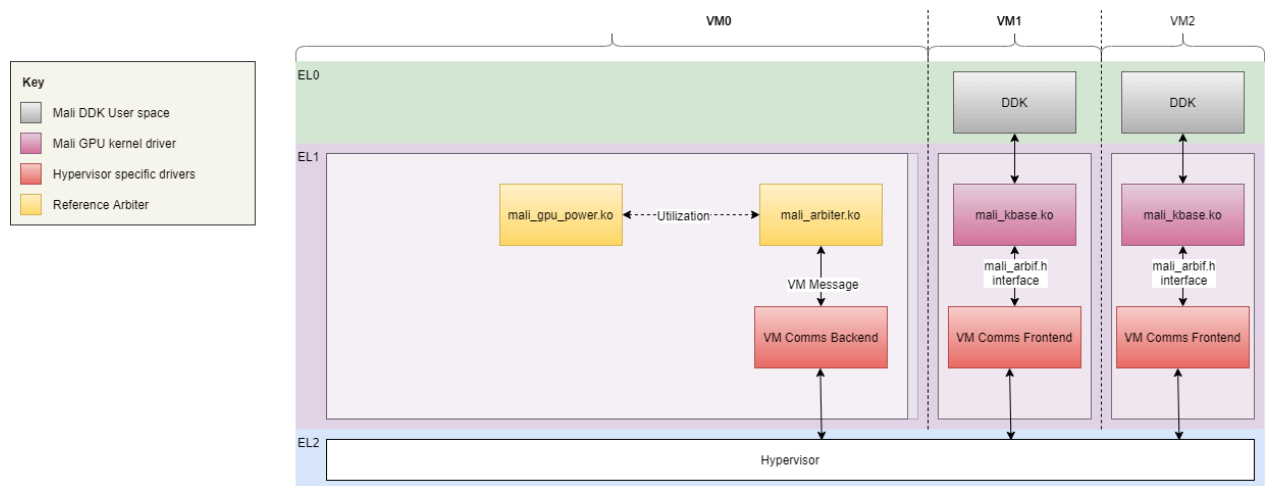
Multiple VMs without dedicated arbiter VM

The following figure shows an example setup with the Xen reference modules.

Figure 2-14: Multiple VMs without dedicated arbiter VM

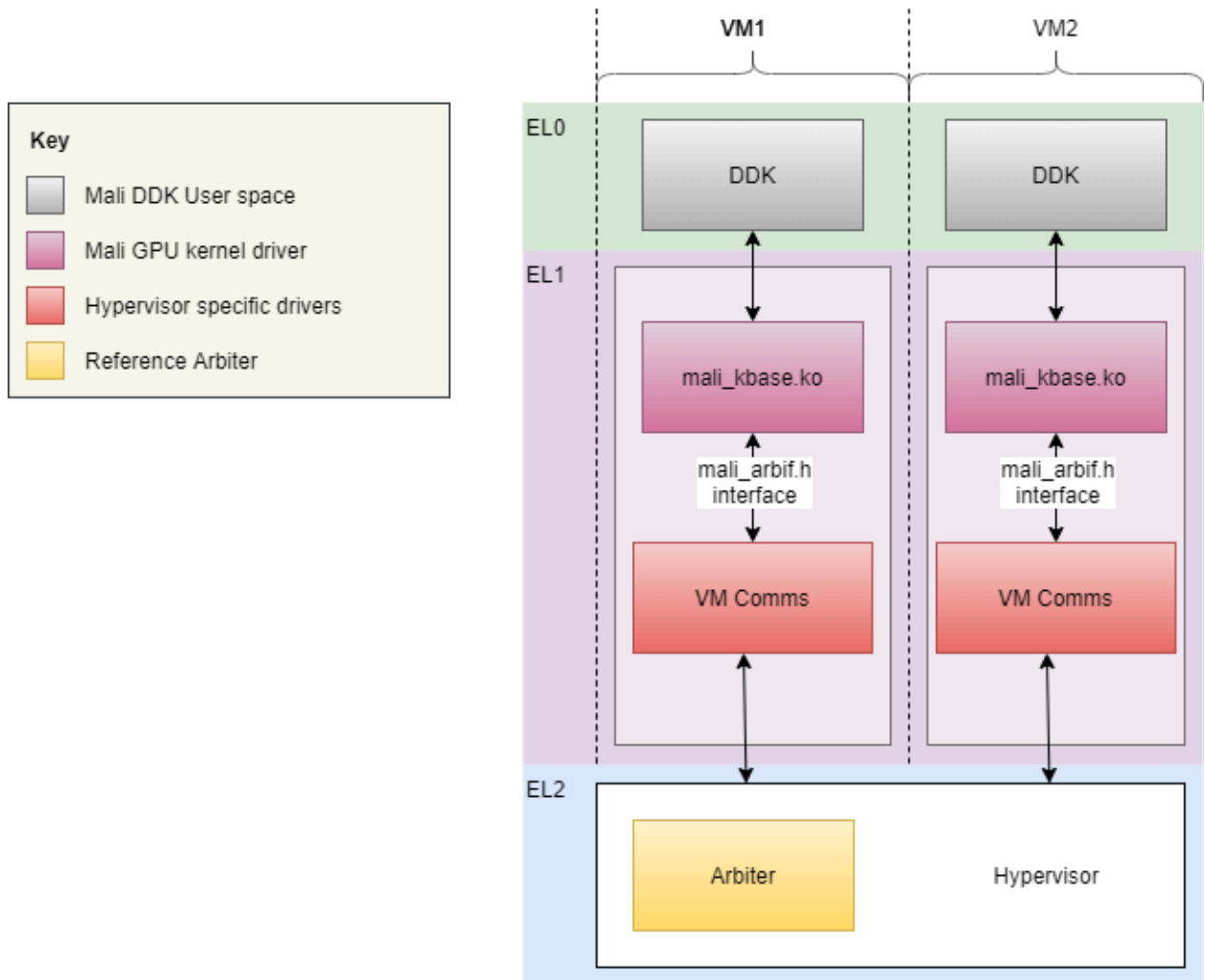
Multiple VMs with dedicated arbiter VM

The figure shows an example with multiple VMs and a dedicated VM for the arbiter.

Figure 2-15: Multiple VMs with dedicated arbiter VM

Arbiter inside hypervisor

The figure shows an example of virtualization with the arbiter inside the hypervisor.

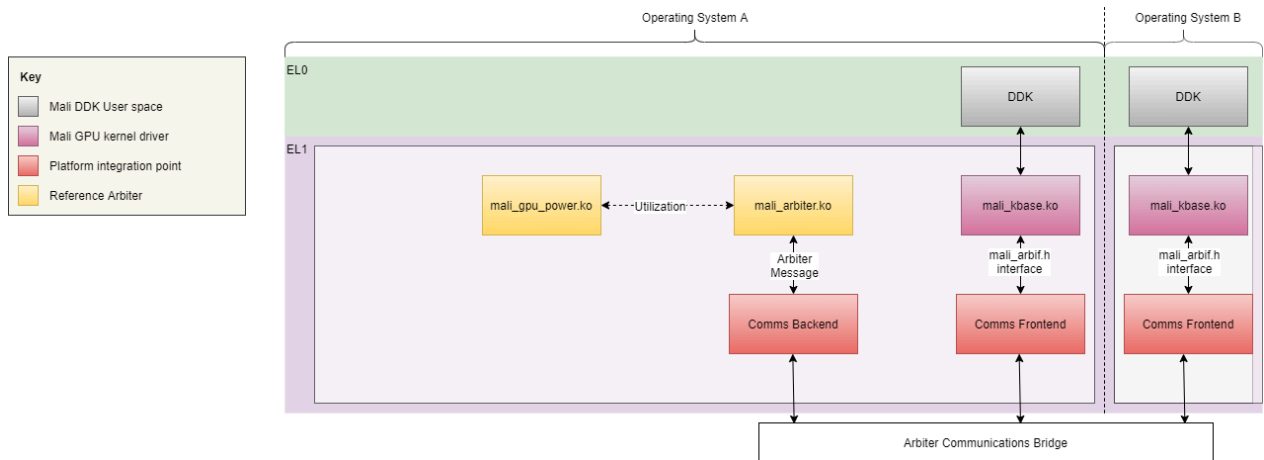
Figure 2-16: Arbiter inside hypervisor

No hypervisor configuration

For the use case where there is no hypervisor there must be:

- A communication bridge between the arbiter and the VMs. This bridge is the mechanism that permits the synchronization between the VMs using the GPU and the arbiter. The arrangement permits the VMs to request the resource when it needs it. The arrangement permits the arbiter to synchronize the access according to its scheduling policy.
- At least one overarching owner of the hardware. This ownership is required to manage the hardware allocation to each VM. It is what actually permits each VM to have direct access to the resources. Therefore, it has to be synchronized with the communication mechanism.

The figure shows an example of a system that runs without hypervisor configuration.

Figure 2-17: No hypervisor configuration

2.7 Reference software stack for paravirtualization

There are different reference software stacks according to the type of virtualization you require. The reference software stack is a modular, where some modules are common while others are specific to a solution.

The modules that make up the reference software stack include, power management, arbiter, and hypervisor blocks.

Common paravirtualization and hardware virtualization functionality

There are common modules for power management, the arbiter, and kernel.

GPU power module

The reference GPU power module implements platform functions for the GPU such as *Dynamic Voltage and Frequency Scaling* (DVFS) and power management. The module is platform-specific and is implemented for each platform requiring support.

Kernel module parameters

There are no module parameters for the GPU power module.

Interfaces

The `mali_gpu_power.h` power module defines power interfaces.

`mali_gpu_power_arbiter_cb_ops`

- The arbiter must provide an implementation for `mali_gpu_power_arbiter_cb_ops` through callbacks that are called by the power module to:
 - Obtain GPU utilization information from the arbiter.
 - Notify the arbiter of frequency changes.

mali_gpu_power_ops

- The power module implements `mali_gpu_power_ops` that is called by the arbiter to:
 - Register itself with the power module, passing in the `mali_gpu_power_arbiter_cb_ops` pointer.
 - Unregister from the power modules.
 - Inform the power module that the arbiter expects the GPU to be powered on.
 - Inform the power module that the arbiter does not need the GPU. Depending on the implementation of the power module, this condition can cause the GPU to be powered off.

The DVFS framework defines some interfaces. The power module implements the `devfreq_dev_profile` interface to:

- Set the target operating frequency.
- Provide current performance status information, including GPU utilization to the DVFS framework.
- Provide the current operating frequency.
- Perform cleanup for a DVFS error or for device removal.

Power management logic

GPU power management

The arbiter has visibility of when VMs require the GPU. The Power Management Module enables the GPU to be powered on or off at the appropriate times. The arbiter can signal to the power module when the GPU is active or idle from the point of view of that arbiter instance. Based on this information, and the activities reported by any other arbiters, the power module can powerdown the GPU when the GPU is not required.

Dynamic Voltage and Frequency Scaling

The arbiter determines the utilization of the GPU across all the VMs using its internal state information. The internal state flags when the GPU is, or is not, using any VM. The VMs send the `REQUEST` and `STOPPED` messages to the arbiter to drive this state. The power module then uses the `mali_gpu_power_arbiter_cb_ops` interface to get the utilization information from the arbiter. The power module then reports the GPU utilization data with a callback on the `devfreq_dev_profile` interface initiated through the DVFS framework. The DVFS framework can then calculate a new target operating frequency and pass this frequency to the power module with another callback on the `devfreq_dev_profile` interface. The power module then changes the GPU clock frequency and notifies the arbiter through the `mali_gpu_power_arbiter_cb_ops` interface. The arbiter in turn notifies the VMs of the frequency when the GPU is granted to them.

Arbiter module

The reference arbiter module schedules shared access to the GPUs from one or more VMs within a system. The arbiter module is platform independent and communicates with other modules through defined interfaces. The arbiter module:

- Coordinate access to the GPU between VMs.
- Uses other components to re-assign the GPU between VMs. The components and processes differ between paravirtualization and hardware virtualization.
- Collects GPU utilization information and informs the Power Management Module when the GPU is required.

The system integrator determines the location of the arbiter within the system. The function of the arbiter does not change with location.

Arbiter module parameters

The arbiter module has these parameters:

request_timeout

The `request_timeout` parameter determines the amount of time each VM is allocated on the GPU when time slicing.

The default value for `request_timeout` is 7ms.

yield_timeout

The `yield_timeout` parameter determines how long the arbiter waits for a VM to yield after being instructed to stop using the GPU. If this time is exceeded, the arbiter removes the GPU from the VM and initiates GPU lost handling.

The default value for `request_timeout` is 7ms.

Arbitration logic

The arbiter works in the following way:

- All VMs that request the GPU are given access to it in the order that they make the requests. This allocation happens in a round-robin fashion. All the VMs have equal priority.
- When a VM has access to the GPU, it is the current machine. When the arbiter receives a GPU idle message from the current machine the arbiter:
 - Immediately sends the current machine a stop message.
 - Hands the GPU over to the next VM.

If there is no other VM waiting, then the GPU is unassigned when the current machine has stopped using it.

- If the arbiter does not receive a GPU idle message from the current machine, then the arbiter can limit the time allocated to the current machine. The maximum time that the arbiter gives a VM to use the GPU is configurable. The configuration happens at runtime during kernel module loading and is the same value for all VMs.
- You can change this time by using the kernel module parameter `request_timeout`. The time value is in milliseconds.
- If the current machine takes too long to respond to the GPU stop request, then the arbiter:

- Forcibly removes the GPU from the address space of the current machine.

The arbiter then assigns the GPU to the next VM. After this operation, the arbiter sends a GPU lost message to the VM that had the GPU removed. You can change the behavior through changes to the time the arbiter gives each VM to respond to the stop request, before sending the `gpu_lost` message. You can configure the value at kernel module load time. To change the time, specify the kernel module parameter `yield_timeout` with a time value in milliseconds.

Arbiter interfaces

The arbiter exposes various interfaces in `mali_arbiter.h`:

- **mali_vm_ops**

The `mali_vm_ops` interface enables VMs to receive messages from the arbiter through a callback and inform the VM:

- To stop using the GPU.
- That the GPU has been granted to it.
- That the GPU is not longer available to it, because the VM took too long to stop using the GPU.

- **mali_arb_ops**

The `mali_arb_ops` interface enables VMs to be registered and to send messages to inform the arbiter that:

- The VM is using the GPU.
- The VM is not currently using the GPU (idle)
- The VM has stopped using the GPU

- **vm_assign_ops**

The `vm_assign_ops` interface enables the arbiter to assign the GPU to a VM. The arbiter uses this interface to assign or unassign the GPU from a VM or to check which VMs are currently assigned to a GPU.

- **arbiter_create and arbiter_destroy**

The `arbiter_create` and `arbiter_destroy` signals enable an alternative mechanism to create and destroy the arbiter instance:

- These signals are not required when the arbiter is instantiated using the probe mechanism.
- The `arbiter_create` function contains the GPU power interfaces, VM assign interfaces and repartitioning interfaces

Arbiter observations

The arbiter does not need access to the GPU or any VM-specific data. It coordinates VM access to the GPU. The arbiter relies on either a hypervisor or a separate GPU control kernel module to change the GPU assignment.

Kernel GPU driver kbase

The kernel GPU driver (kbase) handles requests from the user side library to run content on the GPU. Kbase can be configured to work within a virtualized environment or in nonvirtualized environment.

The kernel GPU driver implements the `arbiter_if_arb_vm_ops` interface to handle communication from the arbiter. kbase has an interface with the module in charge of communication with the arbiter (either the Xen frontend for paravirtualization or access windows for hardware virtualization). The arbiter triggers protocol messages from the communication module on the arbiter side that, in turn, trigger these callbacks. The messages can be on the Xen backend or resource group. This messaging enables kbase to receive:

- Notification from the arbiter to stop using the GPU.
- Notification from the arbiter that the GPU has been granted to the VM.
- Notification from the arbiter that the VM has lost access to the GPU.
- The maximum config from the arbiter.

Device tree binding

The GPU node is defined in the Device tree. The `arbiter_if` phandle must be defined in the GPU node to enable the DDK to use virtualization. This phandle must point to a `mali_arbif` or a `mali-gpu-aw-message` node. If the `arbiter_if` phandle is present but the communication module on the VM side (Xen frontend or access window) is not loaded, then kbase defers probe. If the `arbiter_if` field is not present in the Device tree, then the driver is configured to:

- Work within a non-virtualized environment.
- Assume the GPU is not shared.

Module parameters

There is only one module parameter that is relevant to virtualized environments:

- **gpu_req_timeout**

The `gpu_req_timeout` module operates on a virtualized platform. If the GPU is not granted within this time (ms), kbase defers the probe.

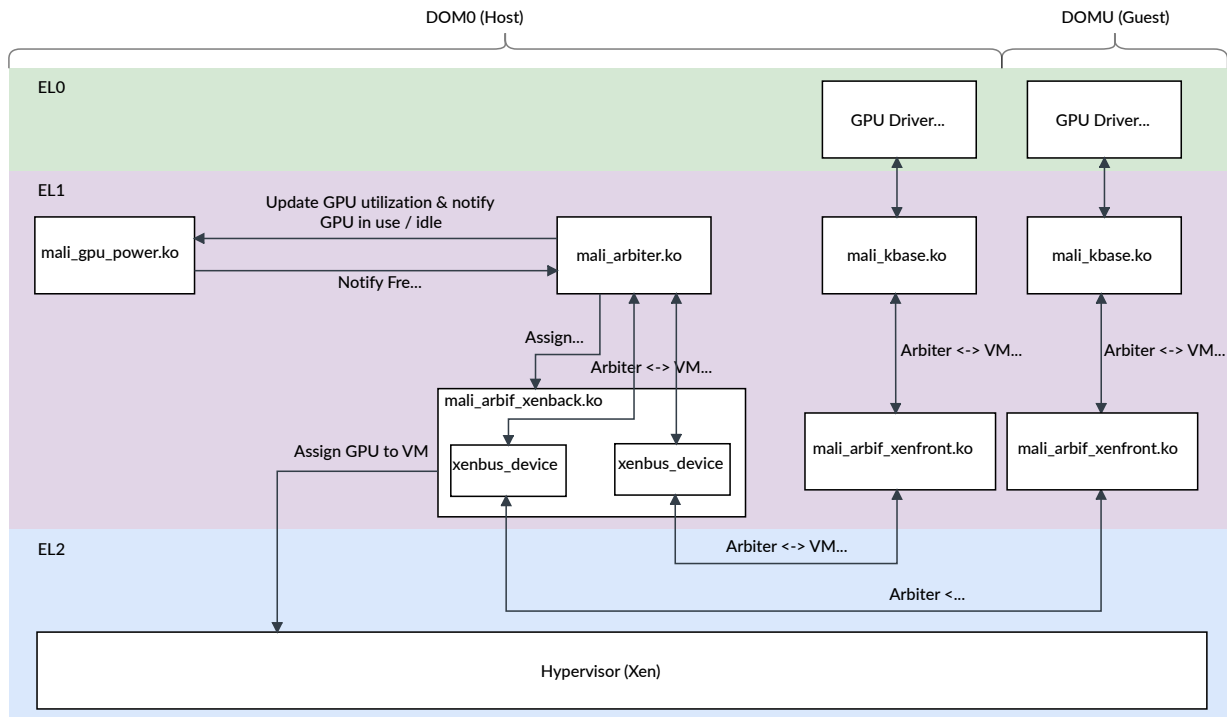
The default for `gpu_req_timeout` is 1ms.

Dependencies

The kernel GPU driver depends on the Xen frontend module to send and receive messages to and from the arbiter through the backend module.

Paravirtualization Reference SW Stack

The diagram shows the modules in the reference paravirtualization software:

Figure 2-18: Reference paravirtualization software modules

There are several separate modules to enable modularity. Some modules are platform-specific, others are platform independent or optional. The modules expose one or more interfaces that permit other modules to call them and receive callbacks.

The modules are:

Arbiter

Filename `mali_arbiter.ko`

The arbiter module schedules VM access to GPU.

The arbiter has some specific paravirtualization features.

Device tree binding

The arbiter is bound to the Device tree. A `platform` field optionally enables the arbiter to locate the GPU power module device when used.

Module parameters

There are no paravirtualization-specific module parameters in addition to the standard parameters.

Dependencies

Power module

The power module is an optional dependency for the arbiter:

- If the platform field is not present, then the arbiter loads without the power module.

- If the platform field is present, then the power module must be loaded before loading the arbiter, otherwise the arbiter probe defers.

Xen backend

The arbiter uses the Xen backend module to send and receive messages to the guest VM:

mali_vm_ops

The `mali_vm_ops` module enables VMs to receive messages from the arbiter with a callback. The arbiter calls handlers in the Xen backend module. The handlers in turn send messages through the Xen bus to the Xen frontend module.

mali_arb_ops

The `mali_vm_ops` module enables:

- VMs to send messages to the arbiter. The Xen backend module:
 - Receives messages from the Xen frontend module through the Xen bus.
 - In turn calls handlers in the arbiter
- The Xen backend module to register and unregister:
 - The VM assign interface, `vm_assign_ops`.
 - VMs

vm_assign_ops

The `vm_assign_ops` module enables the arbiter to assign the GPU to VMs. The Xen backend implements `vm_assign_ops` handlers that are called by the arbiter.

- The handlers make GPU operation hypercalls to the Xen hypervisor.
- The GPU operation hypercalls are a paravirtualization-specific feature added to Xen. For more details, see [B. Xen hypervisor](#) on page 102.

Power

Filename `mali_gpu_power.ko`

The power module enables power up and powerdown of GPU and gets utilization information.

The power module has some specific paravirtualization features and behavior. There is only one arbiter in the system. Therefore when the arbiter informs the power module that it no longer needs the GPU, then the GPU is powered off.

Device tree binding

The power module is enabled and configured with the Device tree.

Module parameters

There are no module parameters.

Dependencies

The power module can be loaded without other paravirtualization modules, however it depends on the arbiter to function correctly.

Xen backend

Filename `mali_arbif_xenback.ko`

The Xen backend module enables the arbiter to communicate:

- With Xen to assign the GPU.
- With the VM (GPU driver) through XenBus.

The Xen backend module provides a link between the arbiter and the Xen hypervisor. When the backend module is loaded, it searches the Device tree for the arbiter and then gets a pointer to the arbiter public device data. This data enables arbiter operations to be called, such as registering domains (VMs) and hypervisor (Xen) callbacks for use by the arbiter.

The Xen backend module also provides a link between the arbiter and the VMs. It sends and receives messages to and from the Xen frontend module using the Xen bus. During protocol initialization handshaking, the Xen backend module checks the protocol version of the Xen frontend module. The version must be equal to or greater than the minimum version supported by the backend.

Device tree binding

The Xen backend module depends on arbiter Device tree node to find the arbiter device.

Module parameters

There are no module parameters.

Dependencies

The backend module depends on the arbiter. If the module is loaded (using `insmod`) before the arbiter is loaded, then probe is deferred.

The Xen backend module also depends on:

- Xen hypervisor to:
 - Communicate with the frontend modules.
 - Assign the GPU.
- Xen frontend module to communicate with the VMs.
- A Xen device connection between the backend and the frontend for each VM that uses the GPU.
 - This connection is achieved by configuring XenStore using the `xenstore-write` tool so that the frontend and backend domains reference each other.
 - The backend module is probed each time a new connection is created.
 - An example of the configuration that would connect the backend on Dom0 with the frontend for Dom1 is shown as follows:

```
# Tell domU about the new device and its backend
xenstore-write /local/domain/1/device/mali_arbif_xen/0/backend-id 0
xenstore-write /local/domain/1/device/mali_arbif_xen/0/backend "/
local/domain/0/backend/mali_arbif_xen/1/0"
# Tell dom0 about the new device and its frontend
xenstore-write /local/domain/0/backend/mali_arbif_xen/1/0/frontend-
id 1
xenstore-write /local/domain/0/backend/mali_arbif_xen/1/0/frontend
"/local/domain/1/device/mali_arbif_xen/0"
# Make sure domU can read the dom0 data
xenstore-chmod /local/domain/0/backend/mali_arbif_xen/1/0 r1
```

```
xenstore-chmod /local/domain/1/device/mali_arbif_xen/0 w1  
# Activate the device, dom0 needs to be activated last  
xenstore-write /local/domain/1/device/mali_arbif_xen/0/state 1  
xenstore-write /local/domain/0/backend/mali_arbif_xen/1/0/state 1
```

Xen frontend

Filename `mali_arbif_xenfront.ko`

The Xen frontend module:

- Enables DDK to communicate with arbiter through XenBus.
- Abstracts Xen related functionality away from the DDK and provides a generic interface to the DDK.

The Xen frontend module provides a link between the kernel GPU driver (kbase) and the arbiter. The module communicates with the Xen backend module through the XenBus. During protocol initialization handshaking, the frontend module checks the protocol version of the Xen backend module. This version must be equal to or greater than the minimum version supported by the frontend.

The Xen frontend module also contains the Arbif adapter. This adapter provides an implementation of the `arbiter_if_vm_arb_ops` interface that enables the kernel GPU driver to use paravirtualization:

- Enables the VM to register and unregister for GPU-related callbacks.
- Get the max config from the arbiter.
- Request GPU access from the arbiter.
- Inform the arbiter that the driver has gone active.
- Inform the arbiter that the driver has gone idle.
- Inform the arbiter that the driver has stopped using the GPU.

Device tree binding

The `mali_arbif` node must be defined in the Device tree.

Module parameters

There are no module parameters.

Dependencies

The Xen frontend module depends on the Xen backend module and the Xen hypervisor to send messages to it (through XenBus)

Module probe sequence

The modules can be loaded in any order but are probed in a certain order due to the dependencies between the modules.

Host module probe sequence

The modules are probed in the following order:

- `mali_gpu_power.ko`
- `mali_arbiter.ko`

- mali_arbif_xenback.ko

VM module probe sequence

The modules are probed in the following order:

- mali_arbif_xenfront.ko
- mali_kbase.ko

Kernel GPU driver

Filename mali_kbase.ko

The mali_kbase.ko module handles requests from a corresponding user side library to perform GPU drawing operations. This module is arbiter aware. It can run independently in a nonvirtualized solution.

Xen

Filename xen.efi

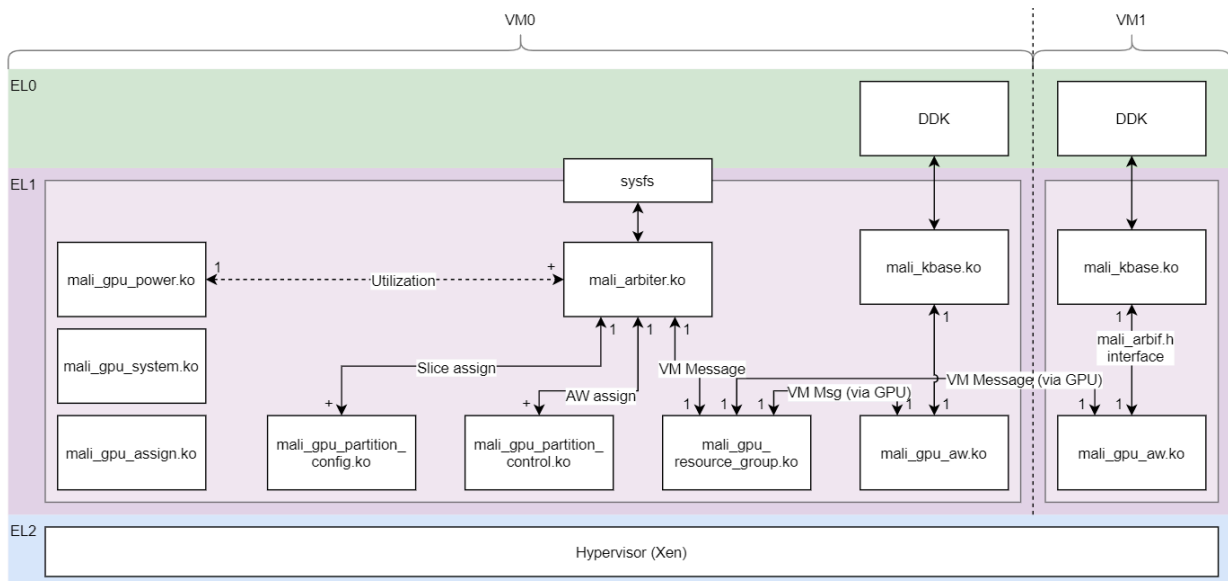
The Xen module is the hypervisor that creates and manages VMs and provides access to hardware, for example. CPU, GPU, and memory.

2.8 Reference software stack for hardware virtualization

The reference software stack for hardware virtualization is modular to enable you to plug and play each module as required. You can choose to use certain modules from the reference implementation and replace other modules with your custom implementation.

The diagram shows the modules in the reference hardware virtualization software.

Figure 2-19: Modules in the reference software stack for hardware virtualization



Each module exposes an interface with platform device data which includes register and unregister functions. These functions permit other modules to register callback functions with this module as required. Each of the modules, except for the arbiter, are platform device drivers and are enabled and configured through the device tree.

Assign

The assign module, `mali_gpu_assign.ko`:

- Assigns resources to resource groups based on the user configuration. In the reference implementation, the user configuration is provided to the module as a command-line argument. Depending on the use case, a separate safety island might handle this configuration.
- Assigns each resource group to an AXI bus to control access.
- Enforces isolation between the partitions in different groups.

System

The system module `mali_gpu_system.ko`:

- Configures normal and protected StreamID of each access window.
- In a debug build, provides fault reporting of the operation of the partition manager.

Partition configuration

The partition configuration module, `mali_gpu_partition_config.ko`:

- Exposes an interface to do slice configuration within the partition.

Partition control

The partition control module, `mali_gpu_partition_control.ko`:

- Exposes an interface to reset the partition and unassign any active access window.
- Exposes an interface to assign an access window to the partition.

Access window

The access window module, `mali_gpu_aw.ko`:

- Sends and receives messages on behalf of a VM to the arbiter (through the resource group).

Resource group

The resource group module, `mali_gpu_resource_group.ko`:

- Creates an arbiter instance.
- Sends and receives messages on behalf of the arbiter to a VM (through an access window).
- Exposes an interface for slice power, clock, and reset control.

Arbiter

The arbiter module, `mali_arbiter.ko`:

- Originates in the resource group and one arbiter exists per resource group.
- Maintains requests from multiple VMs for each partition in the group and time slices the partition between them.

- Does repartitioning and access window reassignment between partitions using the partition configuration and partition control interfaces respectively.
- Controls the slice power, clock, and reset states of slices in the group using the resource group interface.
- Provides utilization information to the power module when requested, and updates the frequency to VMs when it gets notified of change by the power module.

Power

The power module, `mali_gpu_power.ko`:

- Works with the platform Dynamic Voltage and Frequency Scaling (DVFS) module to optimize the GPU frequency based on the utilization reported by multiple arbiters.
- Instantiates sub-sub-nodes and controls module probe order.

Kernel GPU driver

The kernel GPU driver, `mali_kbase.ko`:

- Handles requests from a corresponding user side library to perform GPU operations.

Assign Module

The assign module is a stand-alone entity. When used in the single OS configuration, the assign module configures the assignment of resources to groups and groups to AXI interfaces. The module also configures slice isolation, which prevents adjacent slices in different groups from interfering with each other.

Module parameters

The command-line argument looks like

```
ptm_config='A:S0:S1:S2:P0:P1:W0:W1,B:S3:P2:W2'
```

In this command:

- Comma (,) separates each group.
- Colon (:) separates each entity.
- A or B represents which AXI bus the group is assigned to.
- S represents a slice (ranging from 0-7).
- W represents an access window (ranging from 0-15).
- P represents a partition (ranging from 0-3).

Dependencies

The assign module requires the power module to probe, and can be loaded at any time.

System

The system module configures the normal and protected StreamIDs.

In the reference software stack, the protected and non-protected StreamIDs are statically assigned as follows:

Module parameters

There are no module parameters for the system module.

Dependencies

The system module requires the power module to probe and can be loaded at any time.

The system module receives interrupts for these errors:

AXI_A_PARITY

Error on AXI-A.

AXI_B_PARITY

Error on AXI-B.

AXI_C_PARITY

Error on AXI-C.

GENERAL_HARDWARE

General hardware error.

SLICE_PARITY

Slice parity error.

Range: SLICE0_PARITY to SLICE7_PARITY.

SLICE_ISOLATION

Slice isolation error.

Range: SLICE0_ISOLATION to SLICE7_ISOLATION.

SLICE_BIST

Slice BIST error.

Range: SLICE0_BIST to SLICE7_BIST.

INVALID_BUS

Transaction received for address range that is not valid for that bus.

GROUP_WATCHDOG

Watchdog error for group.

Range: GROUP0_WATCHDOG to GROUP3_WATCHDOG.

PARTITION_LOCKED_ACCESS

Attempt to access a locked partition register.

Range: PARTITION0_LOCKED_ACCESS to PARTITION3_LOCKED_ACCESS.

PARTITION_LOCKED_BIST

Attempt to access a locked BIST controller.

Range: PARTITION0_LOCKED_BIST to PARTITION3_LOCKED_BIST.

PARTITION_MTCRC

Memory transaction CRC error.

Range: PARTITION0_MTCRC to PARTITION3_MTCRC.

AW_INVALID_ACCESS

Attempt to access disabled window.

Range: AWO_INVALID_ACCESS to AW15_INVALID_ACCESS.

In the reference implementation, when an error occurs, no action is taken other than to log the error to dmesg through dev_dbg. You can use this message to debug any related errors if they occur.

The system integrator must take appropriate action according to the use cases they support.

Resource group

The resource group module provides a link between the arbiter and the access windows or VMs. When the resource group module is loaded, it creates the arbiter module using the `arbiter_create()` API. This API is statically exported from the arbiter module. During the creation of the arbiter, the resource group module passes GPU power, VM assign and repartitioning interfaces to the arbiter. This passing enables the arbiter to communicate with the GPU power module and the partition configuration and control modules. In return the resource group gets `mali_arb_ops` for it to call into the arbiter.

After the creation of the arbiter, the resource group registers all the VMs it has access to with the arbiter. This action registers all the access windows assigned to the resource group. During the registration of each VM resource group, process passes the `mali_vm_data` interface to the arbiter, so that the arbiter can communicate with all the VMs.

The resource group is responsible for sending and receiving messages to and from the access window module using the hardware message passing. During protocol initialization handshaking the resource group module checks that the protocol version of the access window module. The version must be equal to or greater than the minimum version supported by the resource group.

The resource group has a complete view of the resources in the group. The resource group enables other modules to query slice, access window, and partition mask data that belongs to this group.

The resource group controls the clock, power, and reset of the slices in this group. The arbiter calls into these APIs to control the slices in the group through `sysfs` user commands.

Module parameters

There are no module parameters for the resource group module.

Dependencies

The resource group module functionality requires:

- The access window module to communicate with the VMs.

The resource group module requires modules for statically exported APIs. These modules must be loaded first:

- Arbiter modules, for `arbiter_create()` and `arbiter_destroy()` APIs.

Modules required for the probe to succeed:

- Power module
- Assign module
- System module

Conditions on which the module probe is deferred:

1. No partitions are assigned to this group. In this case, there is no value in having this module running.
2. Some partitions are assigned but the corresponding partition configuration or partition control device drivers are not yet loaded.

Device tree binding

When used in a single OS configuration, the resource group module parses the Device tree. The module finds the GPU power, the partition configuration and partition configuration nodes corresponding to the partitions assigned to this resource group. The module then retrieves the interfaces implemented by these modules and passes them to the arbiter module in `arbiter_create()` for the arbiter module to access these interfaces. The arbiter uses these interfaces to:

- Communicate with the GPU power module.
- Permit dynamic assignment of resources when the partition configuration and control modules are in the same OS as the arbiter.

Access window

The access window module provides a link between the kernel GPU driver (kbase) and the arbiter. The module communicates with the resource group module with hardware message passing. During protocol initialization handshaking, the access window module checks the protocol version of the resource group module. The version must be equal to or greater than the minimum version supported by the access window.

The access window module contains an implementation of the `Arbif` adapter. This adapter provides an implementation of the `arbiter_if_vm_arb_ops` interface that enables the kernel GPU driver to use hardware virtualization. The adapter:

- Enables the VM to register and unregister for arbiter interface callbacks.
- Gets the max config from the arbiter.
- Requests GPU access from the arbiter.
- Informs the arbiter that the driver has gone active.
- Informs the arbiter that the driver has gone idle.
- Inform the arbiter that the driver has stopped using the GPU.

Module parameters

There are no module parameters for the resource group.

Dependencies

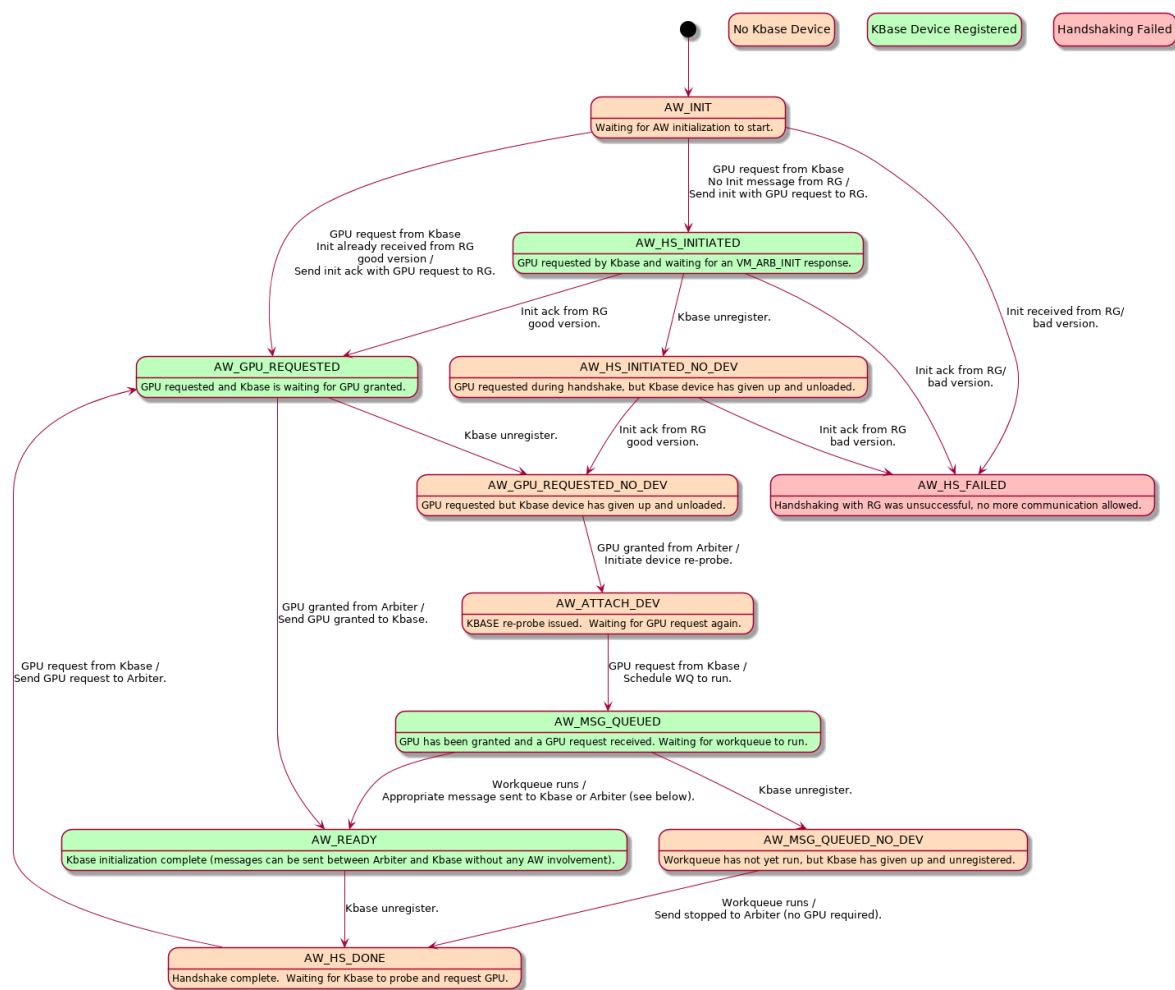
Modules required for functionality:

- Indirectly, depend on the resource group module to send messages to the arbiter.

State transition diagram

The Access Window (AW) module manages communication with the Resource Group (RG) module even when the kernel GPU driver (kbase) is not loaded. The state transition diagram shows the different states and how they interact inside the access window module.

Figure 2-20: State transitions for virtualization modules



Partition Configuration

The partition configuration module configures slices in a partition. The arbiter uses this interface to assign slices to partitions. In the reference implementation, `sysfs` triggers this action.

Module parameters

There are no module parameters for the partition configuration module.

Dependencies

Modules required for the probe to succeed:

- Power module
- Assign module
- System module

Partition control

The partition control module is responsible for resetting a partition and assigning an access window to the partition. The partition control module holds data about which access window is assigned to the partition at any time. The arbiter can query this data when the arbiter restarts a VM.

Module parameters

There are no module parameters for partition control.

Dependencies

Modules required for the probe to succeed:

- Power module
- Assign module
- System module

Arbiter

The arbiter is not a platform device and is not associated with any device tree node. The resource group module creates the arbiter using the `arbiter_create()` API.

The arbiter has some features specific to hardware virtualization. The arbiter module arbitrates partitions between requesting access windows. The arbiter also maintains partition configurations and access window assignments.

slice_power_off_wait_time

Default value 100ms.

The `slice_power_off_wait_time` parameter defines the maximum time (in ms) that the arbiter waits before powering off the slices. This powerdown happens when the slice is unassigned from a partition.

no_timeslice_yield_timeout

Default value 5000ms

The `no_timeslice_yield_timeout` parameter defines the maximum time (in ms) given for a driver to stop before the arbiter resets the GPU. This reset happens when the partition is not time sliced with other drivers.

Dependencies

Modules required for functionality:

Partition control module

The partition control module implements the VM passing interface. The arbiter uses the VM assign interface to assign an access window to a partition.

Partition configuration module

The partition configuration module implements the repartitioning interface. The arbiter uses the repartitioning interface to configure slices in a partition.

Resource group module

The resource group module acts as the parent of the arbiter, creating and destroying it. The arbiter gets the VM assign interfaces and repartitioning interfaces from the resource group module in `arbiter_create()`.

`mali_vm_ops` enables VMs to receive messages from the arbiter through a callback. The arbiter calls handlers in the resource group module which in turn send messages with hardware message passing to the access window module.

`mali_arb_ops` enables VMs to send messages to the arbiter. The resource group module receives messages from the access window module with hardware message passing.

1. The resource group module in turn calls handlers in the arbiter.
2. The arbiter uses resource group public APIs to control power, clock, and reset of slices in the group.

Power module

The power module is optional for the arbiter:

1. If the GPU power device field in the GPU power interface is not present in the `arbiter_create()`, then the arbiter loads without the power module.
2. If the GPU power device field in the GPU Power interface is present in `arbiter_create()` then the arbiter sets the GPU power interfaces internally and finishes the arbiter load.

Sysfs interface into arbiter

The sysfs interface provides a mechanism for the reference implementation to dynamically change certain attributes of the partition manager configuration. Logic specific to the implementation might handle this functionality.

The sysfs interface performs:

- Slice assignment or slice removal to and from a partition,
- Access window assignment or removal from a partition.
- Reading of several hardware attributes.

For the reference implementation of the arbiter, a user-space interface is exposed using `sysfs`. You can configure the partitions using scripting or the Command Line Interface (CLI).

The following example shows the use of the `sysfs` interface to allocate slices 0, 1, 2, and 3 to the first partition. In the example, all literal values are hex bitmasks, where each bit represents a slice or access window.

```
echo "0xF" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/active_slices
```

The following example shows the use of the `sysfs` interface to assign access window 0 to the first partition. In the example, access window 0 has the partition completely to itself with no cooperative sharing.

```
echo "0x1" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/assigned_access_windows
```

The following example shows the use of the `sysfs` interface to assign access windows 0 and 1 to the first partition. In the example, the arbiter time-slices the GPU between them.

```
echo "0x3" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/assigned_access_windows
```

The following example shows the use of the `sysfs` interface to partition an 8-slice GPU into two independent GPUs. The first partition has two slices and a single access window with no cooperative sharing. The second partition has six slices and two access windows which the arbiter can share between them using a round-robin scheduler.

```
echo "0x3" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/active_slices
echo "0x1" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/assigned_access_windows

echo "0xFC" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition1/active_slices
echo "0x3" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition1/assigned_access_windows
```

Repartitioning

Repartitioning is the process of moving slices from one partition to another partition and is achieved through the `sysfs` interface.

Any slices for use in a new partition must not be in use in another partition. A slice must be removed using `sysfs` before reassignment.

Access window reassignment

Access window reassignment is the act of moving an access window from one partition to another partition within the group. This move is achieved through the `sysfs` interface.

Any slices for use in a new partition must not be in use in another partition. A slice must be removed using `sysfs` before reassignment.

Requests from unassigned access windows

The arbiter might receive requests from a VM or access window that is not currently assigned to any partition. When the arbiter receives these requests, it saves the request in a wait list. When the access window is assigned to a partition, the request is moved from the wait list to the request list.

Access window unassignment when the GPU request is in the request queue

A VM or access window can request the GPU but gets unassigned from the partition before being assigned the GPU. The system then moves the request from the request queue to the wait queue.

Slice power hysteresis

The arbiter powers off slices when they are unassigned from a partition and powers them on when they are reassigned. If the delay between the two operations is minimal, the cost of turning the power off and on again is greater than keeping the slices powered on during that time. The arbiter uses a timer to postpone the power off operation until the timeout (`slice_power_off_wait_time`) happens. When the timeout happens, the arbiter turns off the slices. If those slices are assigned to a partition during the count, the timers for those slices are canceled and the slices remain on until they are unassigned again.

Dynamic yield time

Yield time defines how long the arbiter waits for a VM to stop using the GPU after sending a STOP message, and before resetting the GPU. Yield time is a key performance value when a partition is time sliced. Yield time is important because other VMs wait for the GPU until it is taken away from the previous VM. When this VM is the sole assignee of the partition, you can allow more time for the VM to give up the GPU. You can use two module parameters on the command line to configure the time-sliced and non-time-sliced yield times.

- `yield_timeout`
- `no_timeslice_yield_timeout`

Power Module

The power module has some features specific to hardware virtualization. There can be multiple arbiters in the system, one per resource group. Therefore, the power module maintains data specific to each arbiter in a list. The power module:

- Publishes the maximum utilization of all arbiters to the DVFS framework.
- Publishes frequency updates from the DVFS framework to all arbiters.
- Instantiates assign and system devices, and their child nodes.

The power module uses the system and assigns devices on probe. When those devices have probed successfully the power module creates:

- Partition config
- Partition control
- Resource group
- Access window
- Kernel GPU driver

Module parameters

There are no module parameters for the power module.

Dependencies

The power module can be loaded without other hardware virtualization modules. The power module depends on the arbiter to function correctly.

Module Probe Sequence

The modules can be loaded in any order, except that the arbiter must be loaded before the resource group.

Because the power module is responsible for instantiating the subdevice nodes, the power module controls the probe order. The probe order is:

- `mali_gpu_power.ko`
 - `mali_gpu_system.ko`
 - `mali_gpu_assign.ko`

- mali_gpu_partition_config.ko
- mali_gpu_partition_control.ko
- mali_arbiter.ko
- mali_gpu_resource_group.ko
- mali_gpu_aw.ko
- mali_kbase.ko

The arbiter must complete appropriate setup before it can grant the GPU to the kernel GPU driver to complete probing. If the setup is not correct, the probe is deferred. For the kernel GPU driver (mali_kbase.ko) to complete the probe:

- The relevant modules must be loaded.
- The `sysfs` configuration must:
 - Construct a partition and assign slices.
 - Assign corresponding access windows to the partition.

2.8.1 Porting information for other operating systems

We provide high level information to facilitate to porting of the reference stack to alternative *Operation Systems* (OS). You can learn about the functionality of each module in the reference software stack and how it can be modified or used in different scenarios.

The reference software stack is generic and only designed to communicate through the *Graphics Processing Unit* (GPU) channels.

For example, the `sysfs` configuration mechanism to assign slices and access windows to partitions is very flexible for bring up and experimentation, but is not suitable for production.

When modules run in different domains or *Virtual Machines* (VMs), a communication mechanism for the specific integration is required between them. Examples include SCMI, VirtIO, or Mailboxes.

2.8.1.1 System Configuration

The PTM_SYSTEM and PTM_ASSIGN pages are high privilege pages that can affect the operation of all parts of the GPU across multiple domains.



To preserve system security and safety, it is critical that access to these pages is restricted to a privileged or trusted domain.

PTM_SYSTEM

The PTM_SYSTEM page is implemented by the mali_gpu_system.ko module. The PTM_SYSTEM page configures the StreamIDs of each access window. For compatibility with future GPUs, we

recommend keeping this configuration as the default of $2 * \langle \text{AW ID} \rangle$ and $2 * \langle \text{AW ID} \rangle + 1$ for the normal and protected StreamIDs respectively. In a safety critical system, this page is also responsible for fault detection and recovery or reset.

Security and safety

The StreamID page can be used to forcibly reset the resource groups. A malicious user with access to PTM_SYSTEM could modify the StreamID to pretend to be as another VM.

Minimal implementation

The minimum requirement for a minimal implementation is to set the StreamIDs. This implementation is done in `mali_gpu_system.c:gpu_system_probe()` where the PTM_AW[0-15]_STREAM_ID and PTM_AW[0-15]_STREAM_ID registers are written to. This procedure must be executed after every external GPU reset or power on event, for example after device power on or resuming from suspend.

Extra functionality

The reference implementation resets each of the resource groups at probe to allow re-assignment of partitions or slices into different groups. This reset behavior is to enable easier development testing without resetting the system. When a partition has been used, you must reset the resource group before the partition can be re-assigned.

PTM_ASSIGN

The PTM_ASSIGN page is implemented by `mali_gpu_assign.ko` module and assigns resources between buses and resource groups.

The arbiter exchanges data with the preceding slice when configures a slice as a secondary. The arbiter ignores the preceding slice when configures a slice as a primary. An arbiter can incorrectly configure a slice as secondary, rather than primary. In this case the slice might interfere with the adjacent partition. The PTM_ASSIGN_SLICE_ISOLATION register that is under the control of the safety island might incorrectly enforce isolation between partitions.

Security and safety

PTM_ASSIGN is able to modify the slice isolation and assignment of slices and resources to different groups, this is a safety and security concern so this page should be mapped to a privileged domain that can be trusted.

Minimal implementation

For minimal implementation you must configure these registers:

- PTM_RESOURCE_GROUP_BUS
- PTM_PARTITION_RESOURCE_GROUP
- PTM_SLICE_RESOURCE_GROUP
- PTM_SLICE_ISOLATION_SET
- PTM_AW_RESOURCE_GROUP

For the specifications of these registers, see [C.1.2 Register sub-page PTM_ASSIGN](#) on page 129. The configuration is done in `mali_gpu_assign.c:pack_data_into_register_block()`, but this code looks over complicated due to the extra functionality below.

All but `PTM_SLICE_ISOLATION_SET` is self explanatory from the linked register descriptions. For slice isolation, set isolation on Slice 0 resource group, then on the first slice of each new resource group. This provides a hardware barrier to prevent data being able to transfer between adjacent slices in different resource groups.

Extra functionality

To allow easier configuration of resource assignments, the reference implementation provides a module parameter.

In a production system, the system integrator typically sets this module parameter to a static value.

2.8.1.2 Arbiter

The arbiter is associated with the instances of the `PTM_PARTITION_CONFIG`, `PTM_PARTITION_CONTROL` and `PTM_RESOURCE_GROUP` pages. These modules configure the slice allocation, slice power control, and granting of access windows to the partition.

Access to these pages must be restricted to the domain responsible for arbitration. Allowing access to other domains creates a risk of interference to the operation of all access windows within the resource group and therefore to the *Virtual Machines* (VM) under the arbiters control.

2.8.1.3 Re-Partitioning

The `PTM_AW_SET` register in the partition control group determines which window is active for a partition and therefore to which partition a transaction received through that window is sent.

It is illegal to enable the same window in more than one partition or to enable more than one window for a given partition at the same time. An error is reported through `PTM_PARTITION_STATE`.

The steps taken by software to set or change the active slices in a partition use the corresponding `PTM_RESOURCE_GROUP` and `PTM_PARTITION_CONFIG` registers. For manual re-partitioning, the steps are:

1. Disable clocks to the GPU slices being reconfigured by `PTM_SLICE_CLOCK_SET` and `PTM_SLICE_CLOCK_STATE` registers.
2. Assert reset to the GPU slices being reconfigured by `PTM_SLICE_RESET_SET` and `PTM_SLICE_RESET_STATE` registers.
3. Update `PTM_SLICE_MODE_NEW` register.
4. Assert `PTM_SLICE_MODE_UPDATE` register. Hardware will clear this immediately and the register will always read as zero.

5. Poll PTM_SLICE_MODE_ACK register to confirm the change has completed. Hardware indicates an update is in progress.
6. If any of the slices are set to an invalid mode, the error flag in PTM_SLICE_MODE_ACK register is set.
7. De-assert the reset to the GPU slices being reconfigured by PTM_SLICE_RESET_SET and PTM_SLICE_RESET_STATE registers.
8. Enable the clocks to the GPU slices being reconfigured by PTM_SLICE_CLOCK_SET and PTM_SLICE_CLOCK_STATE registers.

2.8.1.4 Time-Slicing

Time slicing is the process of moving the GPU partition between two different VMs. The following steps show how to change which window has access to a partition use the corresponding PTM_PARTITION_CONTROL registers:

1. If the partition is in use, ask the owning driver instance to release it. This step is the arbiter handshake.
2. Reset the partition through PTM_RESET_SET register.
3. Wait for PTM_RESET_STATE to indicate the reset has been applied.
4. Enable a window by setting PTM_AW_SET.
5. Check the window has been enabled without error in the PTM_PARTITION_STATE register.
6. Grant the driver instance access to the GPU. This step is the arbiter handshake.

The steps required for re-partitioning and time-slicing are abstracted into modules for each of the 3 PTM pages that provide interfaces for the functional operations. If you are porting to a different *Operating System* (OS), we recommend reducing the number of interfaces and work queues by combining the arbiter, partition configuration, partition control and resource group into one module.

Arbiter

The arbiter is a software only component and it is implemented by the mali_arbiter.ko module. An instance of the arbiter module is created with each probed resource group module. The arbiter provides a simple round-robin scheduler that fairly schedules each of the access windows requesting access to the partitions in its resource group.

We expect the sysfs configuration interface to be replaced with either system specific static configuration, or an integration specific configuration mechanism.

The hooks into the arbiter are:

- mali_arbiter_core.c: set_slice_assignment()
- get_slice_assignment(), set_aw_assignment()
- get_aw_assignment()

The reference implementation exposes these hooks through a sysfs interface in mali_arbiter_sysfs.c.

The round-robin scheduler has been abstracted into `mali_arbiter_scheduler_rr.c`. This is used as a starting point for an integration specific scheduler.

PTM_PARTITION_CONFIG

The PTM_PARTITION_CONFIG page is implemented by `mali_gpu_partition_config.ko` module and is responsible for configuring which slices are enabled for this partition.

Security and safety

Access to instances of this page must be restricted to the domain or VM owning the assigned arbiter or Resource group.

Minimal implementation

For minimal implementation the code associated this page must configure a partition from the slices available in the resource group.

This is done in `mali_gpu_partition_config.c:mali_gpu_partition_assign_slices()`.



You must set the slice that is furthest to the left or the lowest index SLICE_MODE to Primary. You must set all subsequent slices to Secondary. All slices in a partition must be contiguous.

Minimal implementation sequence:

1. Write the new value to PTM_SLICE_MODE_NEW register.
2. Trigger the update by writing 0x1 to PTM_SLICE_MODE_UPDATE register.
3. Wait for PTM_SLICE_MODE_ACK.STATUS register to clear.



This bit is cleared by the hardware. Do not clear in the software.

4. Check the value of PTM_SLICE_MODE_ACK.ERROR register.

If the error bit is set, then the slice configuration is invalid. Possible causes are slices not contiguous, the slice or slices are in use by another partition.

PTM_PARTITION_CONTROL

The PTM_PARTITION_CONTROL page is implemented by `mali_gpu_partition_control.ko` module. The page is responsible for controlling partition reset and enabling an access window on the partition.

Security and safety

Access to instances of this page should be restricted to the domain or VM owning the arbiter or resource group to which the page is assigned.

Minimal implementation

For a minimal implementation you must provide the arbiter an interface to assign an access window, un-assign an access window, and get the currently assigned access window.

These actions are performed in `mali_gpu_partition_control.c`: `unassign_partition()`, `assign_partition_to_aw()` and `get_assigned_aw()` respectively.

PTM_RESOURCE_GROUP

The PTM_RESOURCE_GROUP page is implemented by `mali_gpu_resource_group.ko` module. The page is responsible for communicating with access windows and controlling slice power and reset.

Security and safety

Access to instances of this page should be restricted to the domain or VM owning the arbiter or resource group to which is it assigned. The resource group has control over slice power and reset, so is able to affect all partitions in the group. There is also a communication channel between each access window and the resource group that could be used to leak data if both sides were compromised.

Minimal implementation

You must provide the arbiter an interface to:

- Get a list of available partitions, slices and access windows.
- Power the slices on and off.
- Enable and disable the slices, clock and reset control.
- Send and receive messages with access windows.

These implementations are done in `mali_gpu_resource_group_main.c`, where the arbiter side of the message protocol is also implemented. For more information about the message protocol, see [3. Message protocol](#) on page 85.



This implementation should be ported in full, unless re-implementing the arbiter to VM protocol which is not recommended.

2.8.1.5 Access Window and Virtual Machine

Each VM requiring a GPU should be assigned an access window.

This is the VMs interface to the arbiter and must be granted access before the VM can use the GPU partition.

PTM_ACCESS_WINDOW

The PTM_ACCESS_WINDOW page is implemented by `mali_gpu_aw.ko` and `mali_kbase.ko`. This page is used for communication to a resource group or arbiter and to allow the DDK driver access to the GPU.

Security and safety

Access to instances of this page should be restricted to the domain or VM that is using this page. The access window has access to the GPU, so allowing access outside of the owning VM would allow data leakage. The PTM_MESSAGE registers allow a communication channel to the corresponding PTM_RESOURCE_GROUP page. This is normally used for arbiter communication, but could be used to leak data if both sides are compromised.

Minimal implementation

For a minimal implementation, you must provide:

- An arbiter messaging protocol.
- Send and receive messages with the resource group.

These implementations are done in `mali_gpu_aw_main.c`, where the VM side of the message protocol is also implemented. For more information about the message protocol, see [3. Message protocol](#) on page 85.



This implementation should be ported in full, unless re-implementing the Arbiter to VM protocol which is not recommended.

2.9 Power management in the virtualization reference stack

The reference software stack for virtualization supports both GPU-wide power control and GPU-internal power control.

The GPU supports power control through:

GPU-wide power control

This control supports:

- *Dynamic Voltage and Frequency Scaling* (DVFS).
- Clock gating and power gating of the whole GPU.

GPU-wide power control is achieved through control of the *Power Management Integrated Circuit* (PMIC), regulator, and clock source external to the GPU.

GPU-internal power control

This control is within a hardware virtualized GPU and supports clock gating or power gating for individual blocks such as slices or cores.

GPU-wide power control

GPU-wide power control includes:

- GPU clock and power gating for powering the GPU on and off.

- DVFS for changing the voltage and clock frequency applied to the GPU.

The GPU supports paravirtualization and hardware virtualization with a single clock and power source.

Arbiters have full visibility of the GPU usage of the VMs under their control. The arbiters report to the GPU power module that is responsible for GPU-wide power control.

The GPU power module provides:

- Well-defined interfaces for GPU-wide power control. These interfaces exchange information with the arbiters, so that the arbiters can remain platform independent. The interfaces permit implementation-specific GPU power management and DVFS implementations with the reference virtualization software. The reference GPU power module uses standard Linux frameworks for both GPU power gating and DVFS.
- Consolidation of the GPU-wide power control information from multiple arbiters. The consolidated information gives a basis for GPU-wide power control decisions.

When using virtualization, global power management in the GPU kernel driver (for example, using DVFS, `runtime_pm`, IPA) must be disabled. This power management must be disabled because power control applies to all VMs and instances of the kernel driver. Therefore the power cannot be controlled by a single kernel driver.

The GPU kernel driver does not load when the corresponding Device tree node contains certain properties related to power management and virtualization is enabled.

The code example shows a combination that results in an error being returned from the probe function of the GPU kernel driver. The error occurs because there is an attempt to use GPU virtualization and also to implement DVFS in the kernel driver:

```
gpu@e0000 {
    compatible = "arm,mali-midgard";
    reg = <0x0 0xe0000 0x0 0x1ffc0>;
    interrupts = <0 168 4>, <0 168 4>, <0 168 4>;
    interrupt-names = "JOB", "MMU", "GPU"; clocks = <&scpi_dvfs 2>;
    clock-names = "clk_mali";
    operating-points = <
        /* KHz uV */
        50000 820000
    >;
    arbiter_if = <&gpu_aw_message_a_0>;
};
```

The kernel driver does not permit a `power_model` child node in the GPU device along with the `arbiter_if` property. This condition indicates an attempt to use IPA along with virtualization. A `#cooling-cells` property in the GPU device indicates that the GPU device is providing control over power dissipation. This form of control is not permitted with virtualization.

You can choose not to implement GPU-wide power control, for example, when there is a limited need to save power. The GPU power module behaves differently according to the platform status whether it is paravirtualized or hardware virtualized. The reference power module determines

whether the GPU is running on a platform with paravirtualization or hardware virtualization. This determination happens at runtime using the Device tree.

GPU clock gating and power gating

Arbiters have full visibility of when their VMs require the GPU and the reference arbiters provide utilization information to the GPU power module. The arbiters communicate this information over a well-defined interface,

When the reference arbiter device is probed, it attempts to register with the power module. When the arbiter is registered, each arbiter device updates the power module when the arbiter becomes active or idle. The idle event only happens when the arbiter has no requests for the GPU.

The partition manager consumes minimal power. When the platform supports it, the GPU power module can powerdown a GPU when not in use.

The reference GPU power module uses a standard Linux `runtime_pm` framework for GPU power gating.

This power management functionality within the reference GPU power module is:

- Optional for paravirtualization.
- Disabled for hardware virtualization.

Paravirtualization

When using paravirtualization, there is a single arbiter. The reference arbiter for paravirtualization notifies the power module when there is no VM requesting the GPU. If no request comes in within 100ms, the power module powers down the GPU using the Linux `runtime_pm`.

Hardware virtualization

For hardware virtualization, guests can send requests to the arbiter for GPU resources through the partition manager. This communication consumes minimal power and does not require an extra message passing channel.

When running on a platform with a hardware virtualized GPU, the reference power module:

- Turns off the GPU after a system suspend request.
- Turns on the GPU after a system resume request.

This power saving Suspend-to-Ram feature only works in a single OS environment with the Linux kernel device link framework. You must also add the `s2r-parent` and `s2r-child` device tree properties to establish the suspend or resume orderings among modules. For more details about the flow of suspend or resume with device tree properties, see `kernel/Documentation/devicetree/bindings/arm/mali-ptm.txt`.

Dynamic Voltage and Frequency Scaling

To avoid the expense of removing and applying power, you can use dynamic control of the clocks and regulators through DVFS.

A single VM, or single arbiter in a multi-arbiter environment, must never be able to unilaterally restrict the performance of the GPU as a whole. For example, a single VM or

arbiter with low usage or frequency requirements must not cause the whole GPU to go into a lower voltage and frequency point.

The reference arbiter and power module use the Linux `devfreq` framework to integrate DVFS control with the platform.

DVFS is enabled when the `clocks` attribute is specified within the reference GPU power module Device tree node. You must specify *Operating Performance Points* (OPPs) in this case. For hardware virtualization, the top-level GPU node is the power module.

When DVFS is enabled, the power module calls each arbiter through a well-defined interface. The Linux runtime PM system drives the call, at intervals requesting current GPU utilization data.

A reference arbiter calculates the utilization figure for the time-sliced resources to which it controls access. This means a single GPU for paravirtualization and one or more partitions for hardware virtualization. The utilization of each resource, GPU, or partition, is calculated based on the time that the resource is active. The calculation determines the proportion of time when the GPU is assigned to a VM and is not idle. This process relies on the VM providing quick notification to the arbiter when it becomes IDLE, to avoid excess power usage. For hardware virtualization where the arbiter controls multiple partitions, the arbiter reports the maximum utilization figure for those partitions. Reporting the maximum values ensures that the system selects the highest required frequency to meet the needs of the most heavily loaded partition.

Hardware virtualization can have several arbiters. When there are multiple arbiters, the power module consolidates utilization from all the arbiters. The power module returns the highest utilization reported by any of its registered arbiters to the kernel.

Based on the utilization data received from all the arbiters, the reference GPU power module can update the GPU clock frequency. A suitable frequency is chosen from the operating points. The GPU power module can also scale the voltage based on demand. The reference GPU power implementation searches for a regulator named `mali` and uses it when it is available. The operating points determine suitable voltages.

The reference software stack does not support arbiters in different domains.

Frequency feedback

For a frequency change, the GPU power module updates all the arbiters with the new frequency. Arbiters must pass on this information to the GPU kernel driver for instrumentation purposes.

The power module notifies the arbiters of frequency changes through the well-defined interface. The arbiter then passes this information to the kernel drivers using the `ARB_VM_GPU_GRANTED` message.

Active VMs are notified immediately, by resending the `ARB_VM_GPU_GRANTED` with the new frequency. The arbiter caches the latest frequency to inform other VMs as they become active.

Power control locations

GPU-wide power control depends on clocks and regulator integration and power policy. To use the GPU-wide power control with your platform, you must integrate the power control with your PMIC, regulator, and clock implementations.

There is a single clock and a single power source to the GPUs. These sources are supported for paravirtualization and hardware virtualization.

GPU power control is only supported within the same CPU cluster as the arbiters. This arrangement simplifies the implementation with the reference software stack.

It is possible to have more complex systems, for example, with multiple CPU clusters or MCU-based power control. System integrators can implement GPU-wide power management with a dedicated microcontroller, *System Control Processor* (SCP), or custom safety Island (using, for example, an R-class CPU). This implementation can use message passing to consolidate utilization or power level information and feed back the frequency in use.

VM

In the reference software stacks for both paravirtualization and hardware virtualization, the arbiter and power modules in a VM implement power control. For example, in `dom0` for a Xen paravirtualized system.

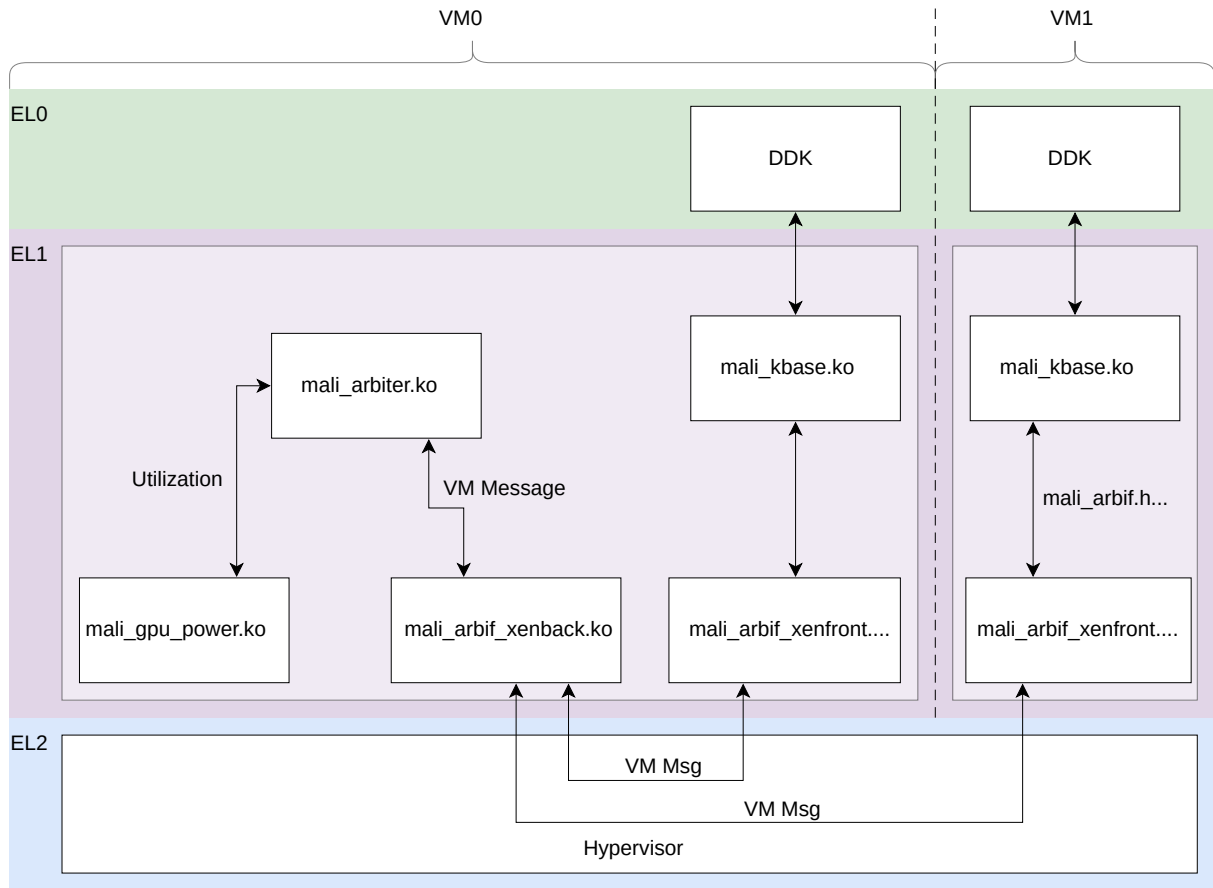
Example 2-1: Paravirtualization with Power Control in a VM

This example has a single CPU cluster and two VMs, one of the VMs is host to the arbiter and GPU power module.

Where power control is implemented within the same CPU cluster as the arbiter, the power module can control the power directly.

You can use the reference power module with Linux (`runtime_pm/devfreq`) frameworks by adding their clock and regulator devices to the Device tree. You can choose to change the governor or sampling period in this case.

Alternatively, you can integrate your own GPU power gating and DVFS implementations with the reference virtualization software. Implement your own GPU power module with the defined arbiter interfaces. You can implement a use-case driven power policy rather a utilization-based one in the power module.

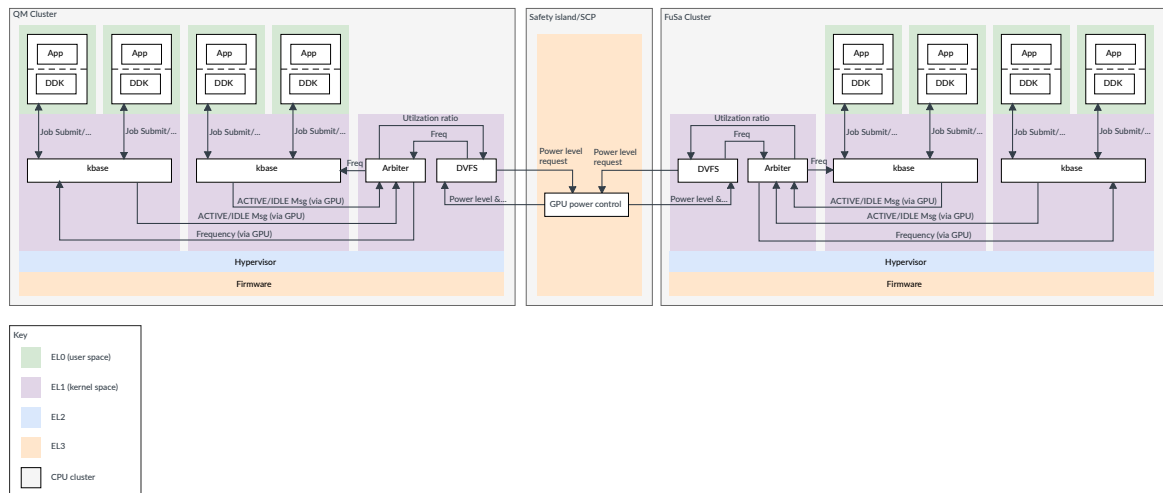
Figure 2-21: Power control for VMs with paravirtualization**Hardware virtualization with power control in a safety island or SCP**

A system can have multiple compute clusters, with the power control delegated to either a safety island or a dedicated micro-controller (SCP). The diagram shows such a system, and the data flows around it, that you must implement.

The hardware virtualization features provide the data flows between the arbiter and kbase, within a single CPU cluster. You must implement the data flows between the clusters, to pass power level requests from the power modules of the arbiters.

The distinction between utilization and power level in the diagram:

- Utilization is a proportion of the active time since the last value.
- Power level is a specific GPU operating point.

Figure 2-22: Power control for the GPU with virtualization

Cross-domain communication

Multiple compute clusters must implement cross-cluster message passing, to:

- Consolidate utilization or power level information.
- Feed back the frequency in use.

Each arbitrator can directly provide a utilization value, but this method requires frequent messages to the safety island or SCP. Communicating changes to the power level might result in fewer messages and therefore a simpler SCP.

Passing power levels between domains requires a protocol that can communicate which power levels exist. Alternatively this information must be known in advance (for example, to be compiled in).

Considerations:

- Transport. How messages are passed between the domains, for example, through shared memory or by dedicated hardware messaging.
- Protocol. How the messages are structured.

You can use the *System Control and Management Interface* (SCMI) as a standard protocol for this communication. In SCMI protocol, each arbitrator is an agent and communicates with the SCP or safety island with some combination of:

- The power domain management protocol.
- The performance domain management protocol.

When using the performance domain management protocol, there is no per-performance level frequency. The frequency can be derived for the current power level based on the `sustained_perf_level` and `sustained_freq` attributes from the `PERFORMANCE_DOMAIN_ATTRIBUTES` message. The performance levels are guaranteed to be a linear scale.

If the arbitrator is run on Linux, then there is an implementation of SCMI that can be used (and includes the calculation of frequency).

GPU-Internal Power Control for slice control

In hardware virtualization, it is possible to reduce power through slice-level control. In complex systems with multiple arbiters, it is important to not rely only on global power control to maximize power savings. You can implement DVFS to scale the performance of the GPU, but it might not be possible to scale the systems down significantly. One or more partitions might be kept at a sustained load. Consider a system that supports both infotainment and digital cockpit use cases. The digital cockpit use case might be safety critical and the infotainment use case not safety critical. While the infotainment load might vary significantly depending on use, the cockpit load is likely to remain reasonably constant. If the GPU is partitioned to separate these two use-cases, the GPU operating point might have to remain high. In such cases, you can reduce the power by ensuring slices that are not needed are powered down. The arbiter performs the powerdown for each resource_group that owns them. In such systems, DVFS might not be worthwhile.

In complex systems, it is unlikely that the GPU becomes completely idle, and therefore implementing power control for the GPU might not be useful.

In the reference software stack, the arbiter implements powering down individual slices:

Powering down slices in an idle partition

The reference arbiter powers down all the slices in an idle partition. A hysteresis timer in the GPU kernel driver:

- Prevents inadvertent powering down during brief pauses in activity, for example, while waiting for `vsync`.
- Permits power down for more significant pauses, for example, when the system is completely idle or waiting for several seconds for user input.

Powering up the slices takes time, which reduces performance briefly. The timer ensures this activity is infrequent and only after a period of activity when a delay is generally acceptable.

Powering off slices not in a partition

You can unassign slices from a partition using the well-defined `sysfs` interface. The reference arbiter powers down a slice when unassigned, if it is not assigned again after a hysteresis timer passes. The length of this timer is configurable using an arbiter module parameter.

You can therefore modify GPU-internal power control using the `sysfs` interface to assign and unassign slices.

2.10 Build reference software for virtualization

To use any form of virtualization with GPUs, you must set up the environment for your Board Support Package to include all the necessary software components.

To deploy the reference virtualization software, you must:

- Build the reference code.
- Build the *Driver Development Kit* (DDK) source code.
- Download and patch the hypervisor source code if using a paravirtualization reference stack.

Some parts of the build process for the DDK are described in the *Arm® GPU DDK Integration Manual*.

Build the reference code

The reference virtualization source code is released separately to the DDK in one or two archives. See the *Release Note* for the exact archive names for your product.

The archive contains the reference code for the arbiter and all the other modules, including the Xen hypervisor-specific ones. Xen is an open-source project, freely available online.

Install source code

To install the reference virtualization source code:

1. Extract the non-virtualized DDK tarballs. When the DDK is extracted, it automatically creates a `driver` directory.
2. Locate the `driver` directory in your filesystem.
3. Copy the archive into the `driver` folder.
4. Extract the archive into the `driver` folder.

```
tar -xvf <archive file>.tar
```

5. Check that the main `driver` folder contains:
`driver/product/kernel/drivers/gpu/arm/arbitration/arbiter`

`driver/product/kernel/drivers/gpu/arm/arbitration/include`

`driver/product/kernel/drivers/gpu/arm/arbitration/power`

`driver/product/kernel/drivers/xen`

Build the DDK source code

You must configure the build environment according to the *Arm® GPU DDK Integration Manual*. Following those procedures makes the Linux kernel directory available.

The reference code build is fully integrated with the normal build system of the DDK and therefore uses the Bob build system.

You can compile all the reference kernel modules together with the standard kernel source code for the DDK.

Building the kernel out of tree

When required, you can build the Linux kernel objects separately, using Bob.

To build the kernel objects separately, using a `bash` environment:

```
cd driver/product
export BUILDDIR=<path_to_your_dir>/build
bldsys/bootstrap_linux.bash
$BUILDDIR/config/settings
  KERNEL_DIR=<your_linux_kernel_root>MALI_ARBITER_SUPPORT=y
  MALI_HAS_ARBITER=y BSP_HAS_HYPERVISOR=y MALI_PARTITION_MANAGER=n
$BUILDDIR/buildme
```

Where:

settings

Sets the options for the DDK, see the Build options section in the *Arm® GPU DDK Integration Manual*.

KERNEL_DIR

Defines the Linux kernel root with which to compile the kernel modules.

MALI_ARBITER_SUPPORT

Enables the standard DDK to support virtualization.

MALI_HAS_ARBITER

Enables the build of the reference arbiter modules (`mali_arbiter.ko` and `mali_gpu_power.ko`)

MALI_PARTITION_MANAGER

It is required for hardware virtualization. If you use it for paravirtualization, it might slow down the build time.

BSP_HAS_HYPERVISOR

Enables the build of the Xen hypervisor-specific modules (`mali_arbif_xenback.ko` folder and `mali_arbif_xenfront.ko`). This option is enabled by default if your Linux kernel configuration file has `CONFIG_XEN` enabled. It in turn enables the previous two configurations. This option is not required for a minimal host-only setup for hardware virtualization.

The default location for the output of the compilation is `$BUILDDIR/install/modules`. If you redefined the `INSTALL_PATH` and the `MODULE_PATH` variables, then the output is in the corresponding path.

If you want to build the Linux kernel objects separately, using the Linux kernel build system, then run the following commands for each folder (module) of the `arbitration` directory:

```
cd driver/product/kernel/drivers/gpu/arm/arbitration/<module>
ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- KDIR=<your_linux_kernel_root>
make
```

Building the kernel in the tree

The kernel module directories are organized so that you can overlay them onto a Linux kernel source tree. This overlay enables in-tree module building and statically linking the drivers into a kernel.

After you overlay the kernel module directories into your kernel tree, you must modify the build to enable the kernel to call the new modules.

You must complete the following steps to link the DDK and the reference modules to the Linux kernel build tree. These steps are also described in the *In-tree kernel build* section in the *Arm® GPU DDK Integration Manual*.

1. Add the `arm` directory to the list of directories in `drivers/gpu/Makefile`:

```
obj-y += drm/ vga/ stub/ arm/
```

2. Add the following line to `drivers/video/Kconfig` after adding it after the line `source "drivers/gpu/ion/Kconfig"`:

```
source "drivers/gpu/arm/Kconfig"
```

3. To build the arbitration modules for virtualization, the arbitration `Kconfig` file must be sourced. Add the following line to `drivers/gpu/arm/midgard/Kconfig` before the last `endif`:

```
source "drivers/gpu/arm/arbitration/Kconfig"
```

4. To build the partition manager modules for hardware virtualization, the PTM `Kconfig` file must be sourced from the arbitration `Kconfig` file. Add the following line to `drivers/gpu/arm/arbitration/Kconfig` before `source "drivers/xen/arm/Kconfig"`:

```
source "drivers/gpu/arm/arbitration/ptm/Kconfig"
```

To compile the reference modules:

1. Enable `MALI__ARBITRATION` for building the power and arbiter modules.
2. Enable `MALI__XEN` for building the XEN related modules.
3. Enable `MALI__PARTITION_MANAGER` for building partition manager modules required by a hardware virtualized GPU.

To configure the kernel and proceed with the build process, see and continue from *Platform-specific configuration file*, described in the *In-tree kernel build* section in the *Arm® GPU DDK Integration Manual*.

Download and patch Xen hypervisor source code

The reference implementation is built around the Xen hypervisor. You must build and integrate the hypervisor into the Board Support Package of your platform.

Xen is an open-source project. For more details about the project, and how to build and integrate the hypervisor in your platform, see the [Xen Project](#).

The Xen version used in this reference implementation is v4.14 for both paravirtualization and hardware virtualization. This version is modified to support Mali™ virtualization and patches are available in this directory: `driver/product/kernel/drivers/xen/arm/scripts/patches`.

To download and patch Xen with the necessary modifications:

1. To store Xen, create a folder:

```
mkdir <base_folder>
```

2. Copy the content of `driver/product/kernel/drivers/xen/arm/scripts`.

3. To get and patch the source code, run the script:

```
cd <base_folder>
chmod +x
./download_and_patch.sh <virtualisation type>
```

For example:

```
download_and_patch pv
```

or

```
download_and_patch hv
```

At the end of the procedure, <base_folder>/host/xen/ contains a fully patched version of the Xen hypervisor. This version is compatible with the rest of the reference modules and ready to be compiled and integrated in your Board Support Package.

2.11 Software debug features with paravirtualization

The reference virtualization software provides various debugging features.

The software includes mechanisms for debug and traces that can help to resolve crashes and lock-ups.

Missing GPU granted message

There are some scenarios that prevent kbase in a VM ever receiving a `ARB_VM_GPU_GRANTED` message. Kbase might not receive a granted message after repartitioning, where GPU access for the VM is removed, or the arbiter fails. When the granted message is not received, all the jobs in kbase are stuck waiting for the GPU to be available.

If kbase does not receive a GPU granted message, it prints a warning message in dmesg:

```
Still waiting for GPU to be granted from Arbiter after 1000 ms
```

Arbitration debugfs traces

Events are available to aid debugging through the ktrace mechanism. These trace events can help when complex crashes or lock-ups happen. You can access the trace through the `/sys/kernel/debug/mali0/mali_trace debugfs` node by, for example, using the command

```
# cat /sys/kernel/debug/mali0/mali_trace
```

ARB_GPU_LOST

`ARB_GPU_LOST` is logged when GPU lost handling starts and when GPU lost handling is complete. The information value field indicates the GPU lost status:

0	GPU LOST clear
1	GPU LOST

For example:

```
<secs>,<tid>,<cpu>ARB_GPU_LOST,,,,,0x0000000000000001 GPU LOST
<secs>,<tid>,<cpu>ARB_GPU_LOST,,,,,0x0000000000000000 GPU LOST clear
```

ARB_VM_STATE

`ARB_VM_STATE` is logged whenever a VM state changes. The information value field indicates the new state:

0	KBASE_VM_STATE_INITIALIZING
1	KBASE_VM_STATE_INITIALIZING_WITH_GPU
2	KBASE_VM_STATE_SUSPENDED

3	KBASE_VM_STATE_STOPPED
4	KBASE_VM_STATE_STOPPED_GPU_REQUESTED
5	KBASE_VM_STATE_STARTING
6	KBASE_VM_STATE_IDLE
7	KBASE_VM_STATE_ACTIVE
8	KBASE_VM_STATE_SUSPEND_PENDING
9	KBASE_VM_STATE_SUSPEND_WAIT_FOR_GRANT
10	KBASE_VM_STATE_STOPPING_IDLE
11	KBASE_VM_STATE_STOPPING_ACTIVE

For example:

```
<secs>,<tid>,<cpu>ARB_VM_STATE,,,,,,,,0x0000000000000006 the VM is entering
KBASE_VM_STATE_IDLE state
<secs>,<tid>,<cpu>ARB_VM_STATE,,,,,,,,0x000000000000000a the VM is entering
KBASE_VM_STATE_STOPPING_IDLE state
```

ARB_VM_EVT

ARB_VM_EVT is logged when kbase receives an event from a VM. The information value field indicates the event code:

1	KBASE_VM_GPU_INITIALIZED_EVT
2	KBASE_VM_GPU_STOP_EVT
3	KBASE_VM_GPU_GRANTED_EVT
4	KBASE_VM_GPU_LOST_EVT
5	KBASE_VM_GPU_IDLE_EVENT
6	KBASE_VM_REF_EVENT
7	KBASE_VM_OS_SUSPEND_EVENT
8	KBASE_VM_OS_RESUME_EVENT

For example:

```
<secs>,<tid>,<cpu>ARB_VM_EVT,,,,,,,,0x0000000000000005 the VM has sent
KBASE_VM_GPU_IDLE_EVENT to the arbiter
<secs>,<tid>,<cpu>ARB_VM_EVT,,,,,,,,0x0000000000000002 the VM has sent
KBASE_VM_GPU_STOP_EVT to the arbiter
```

System module IRQ messages

The system module logs:

- Bus parity
- Partition watchdog
- Slice isolation
- CRC
- General hardware errors.

These events are reported through the deferred and uncorrected error interrupts of the system module. The following type of entry can be seen in the dmesg log at debug level, for example:

```
Detected UNCORRECTED_IRQ_AXI_A_PARITY
Finger print of the error: 0x01234567
```

The fingerprint of the error provides information indicating its cause.

Partition manager version message

The Unsupported partition manager version can be reported in the dmesg output. This message indicates that the Device tree is configured to use a resource that was not assigned to that group or bus. For generic configurations, this message is not cause for concern. Some specific configurations do not expect the assign module to move around resources. For these configurations, this message might indicate an incorrect DTB file or assign module configuration.

For example, in a configuration with no AXI-B in the device tree, an attempt to create a resource group on AXI-B using:

```
insmod mali_gpu_assign.ko ptm_config='B:S0:P0:W0'
```

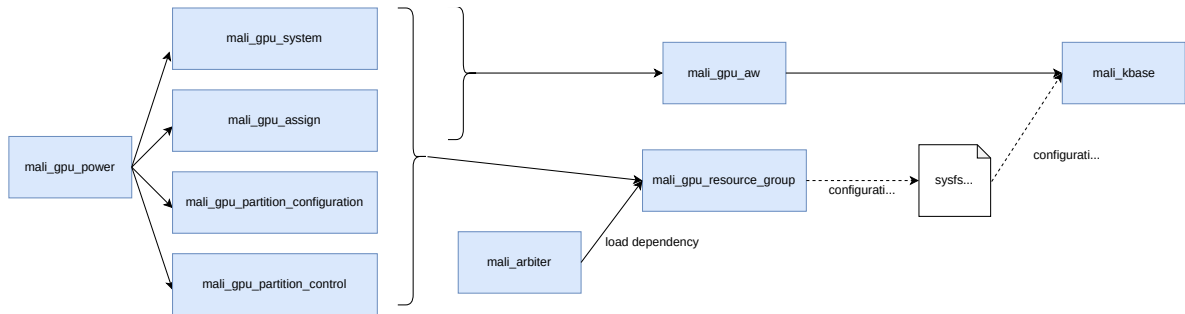
results in the following dmesg output:

```
mali_gpu_assign 6e810000.gpu_assign: Resource Group Assignment:-
RG0 BUS[B] S[0] P[0 1 2 3] W[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15]
RG1 BUS[A] S[ ] P[ ] W[ ]
RG2 BUS[A] S[ ] P[ ] W[ ]
RG3 BUS[A] S[ ] P[ ] W[ ]
mali_gpu_assign 6e810000.gpu_assign: Probed
mali_pm_config 6e020000.gpu_partition_config: Unsupported partition manager
version.
mali_pm_config 6e040000.gpu_partition_config: Unsupported partition manager
version.
mali_pm_config 6e060000.gpu_partition_config: Unsupported partition manager
version.
mali_pm_config 6e080000.gpu_partition_config: Unsupported partition manager
version.
```

This error is also seen when the hardware virtualization reference stack is used with an incompatible GPU.

Dmesg output and probe and load dependencies

Reference arbitration modules can be loaded in any order, with the exception of `mali_gpu_resource_group` which requires the `mali_arbiter` to have been loaded. There are probe order dependencies which mean that some modules defer probe until those dependencies are met.

Figure 2-23: Partition manager probe and load dependencies**Loading any other reference arbitration module before mali_gpu_power**

mali_gpu_power acts as a bus controller and has the responsibility to probe drivers. So, any reference arbitration module which is loaded before mali_gpu_power is not probed and no dmesg output is seen.

Loading mali_gpu_resource_group before mali_arbiter

mali_gpu_resource_group imports symbols from mali_arbiter and must therefore be loaded after mali_arbiter. If it is not, then the following error appears in the dmesg output

```
mali_gpu_resource_group: Unknown symbol arbiter_create (err -2)
mali_gpu_resource_group: Unknown symbol arbiter_destroy (err -2)
```

Loading mali_gpu_resource_group before mali_gpu_partition_control and mali_gpu_partition_config

Loading mali_gpu_resource_group before mali_gpu_partition_control and mali_gpu_partition_config causes it to report the following errors and defer probe:

```
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Partition driver not available.
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Resources get failed
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Arbiter initialization failed
```

When mali_gpu_partition_control and mali_gpu_partition_config are loaded, mali_gpu_resource_group probes successfully:

```
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Arbiter running with Partition Manager
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Arbiter Probed
mali_gpu_resource_group 6e0a0000.gpu_resource_group: Probed
```

Loading mali_kbase before mali_gpu_aw

The access window provides mali_kbase with an interface to the arbiter. Without it, mali_kbase reports an error and defers:

```
mali 6e0e0000.gpu: arbiter_if driver not available
mali 6e0e0000.gpu: Failed to initialise arbif module
mali 6e0e0000.gpu: Device initialization Deferred
```

Loading mali_kbase before configuration

Without a configuration, the arbiter is unable to route a GPU_REQUEST from mali_kbase to a partition. Therefore loading mali_kbase without a configuration causes it to report an error and defer:

```
mali 6e0e0000.gpu: Arbitration interface enabled  
mali_gpu_resource_group 6e0a0000.gpu_resource_group: GPU_REQUEST from VM 0  
not assigned to a partition  
mali 6e0e0000.gpu: Device initialization Deferred
```

Providing a configuration then allows mali_kbase to probe.

Debug Level

Changing the kernel debug level to 7 (debug) produces more verbose dmesg output that is useful for debugging the reference code:

```
echo 7 > /proc/sys/kernel/printk
```

2.12 Protected content with paravirtualization

You can implement secure memory operation with paravirtualization.

TrustZone® Media Protection (TZMP) provides an architecture that restricts access to reserved memory regions. TZMP provides effective memory protection, and is relatively simple to integrate and deploy.

TZMP protects media content from unauthorized access by preventing components running:

- In normal mode reading from a protected memory.
- In protected mode from writing to unprotected memory.

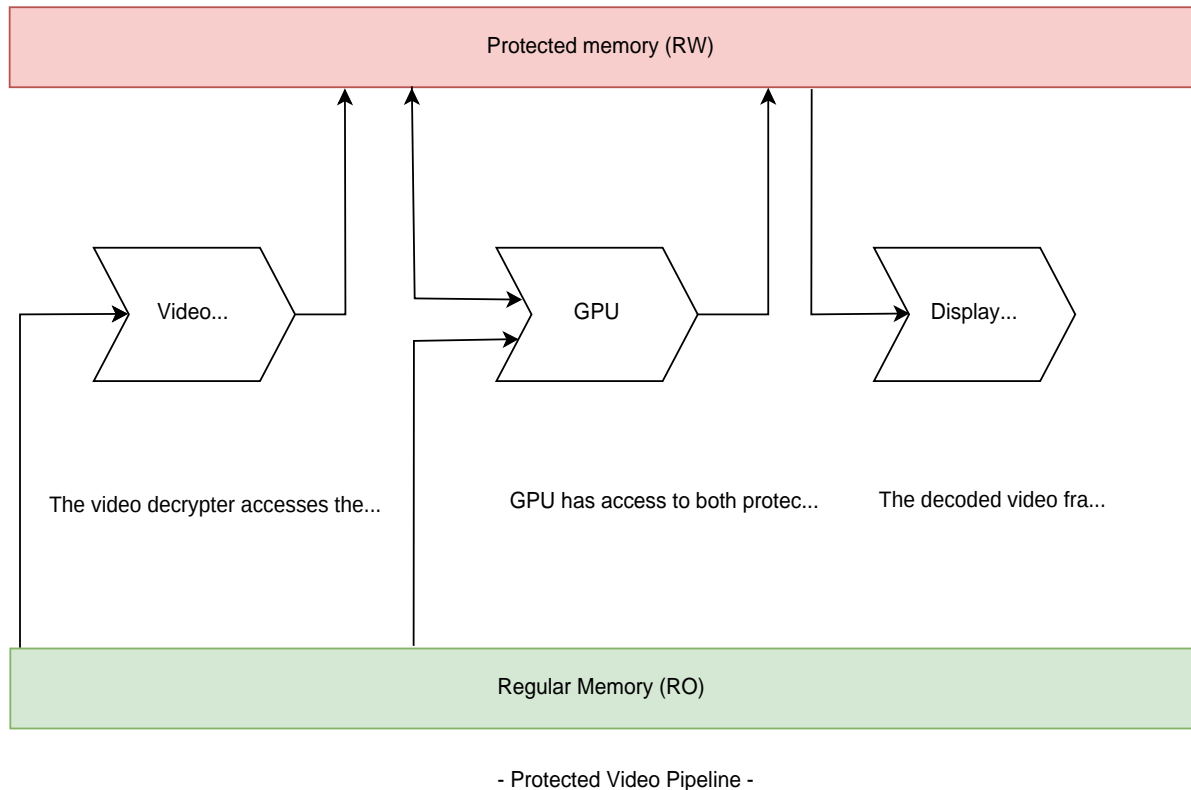
TZMP uses *Non Secure Access IDs* (NSAIDs), and access rules:

- NSAIDs:
 - NSAIDs identify bus requesters.
 - Each transaction is tagged with an NSAID which provides the TZMP firewall with the context to check the permissions.
 - NSAIDs are signals conveyed alongside requests from bus requesters.
 - For GPUs before the introduction of the AMBA® 5 AXI protocol, GPUs use NSAIDs to tag memory transactions. GPUs that use AMBA® 5 AXI protocols, and later, use STREAMIDs to tag memory transactions.
 - A bus requester is usually assigned two NSAIDs for accessing protected and non-protected memory regions.
- Access rules:
 - The TZMP firewall enforces access rules on the client side for TZMP1.

- Checks occur between the system bus and the memory.

The diagram illustrates an implementation of a protected media pipeline using TZMP capabilities.

Figure 2-24: Protected video pipeline

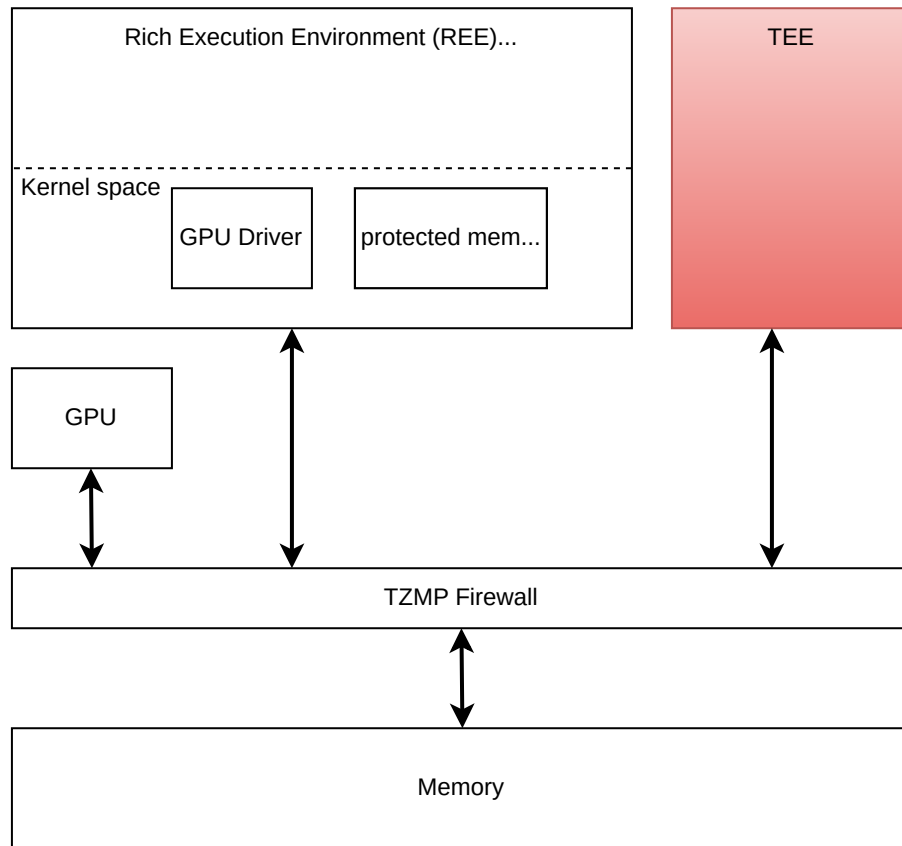


TZMP Software architecture

Secure firmware running at boot time configures the TZMP firewall. The firmware:

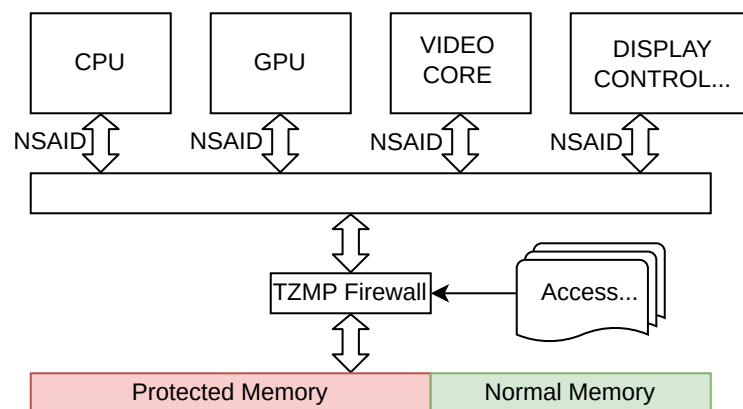
1. Assigns NSAIDs to bus requesters. TZMP1 supports up to 16 NSAIDs.
2. Defines the memory layout:
 - Start physical address of each region
 - Size of memory regions
 - TZMP1 supports a limited number of regions (up to 8 programmable regions and one non-programmable for TZC-400)
 - TZMP1 constrains the regions to be contiguous
3. Defines the access rights for each memory region

The diagram shows the software architecture.

Figure 2-25: TZMP1 software architecture

- TZMP1 Software Architecture -

The diagram shows the inclusion of a firewall.

Figure 2-26: TZMP1 firewall architecture

-TZMP1 Firewall architecture-

TZMP Integration

The main areas to consider when integrating TZMP1 in a paravirtualized or hardware virtualized solution are how to propagate:

- The memory layout defined by the secure firmware to each Virtual Machine (VM) that needs access to protected memory regions.
- The roles defined by the secure firmware to each bus requester.

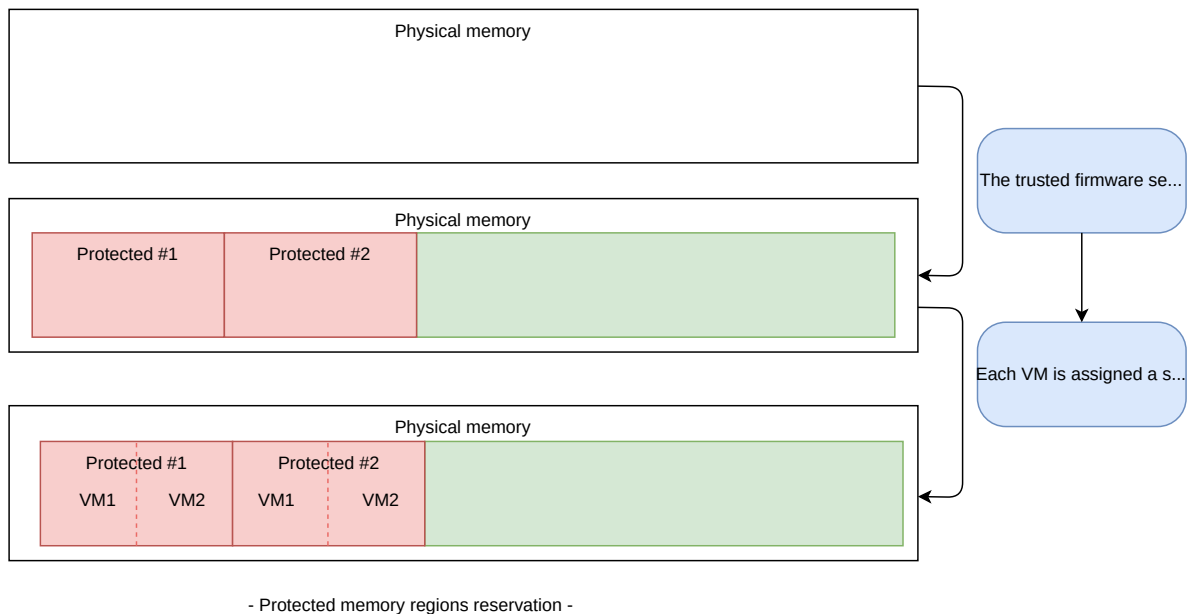
Roles and NSAIDs are attached to hardware bus requesters, so the system behaves identically independent of whether it is a PV, HWV, or non-PV system. For a PV system, the GPU always emits the same assigned NSAID regardless the VM it is assigned to. For an HWV system, the reference software stack can configure each access window and StreamID (and resulting NSAID). This configurability permits the relevant access window to distinguish the different VMs.

Memory regions management

The reference implementation assumes:

- The number of VMs is known.
- Each VM is assigned a fixed amount of protected memory. A static partitioning of memory gives a simpler implementation.

Figure 2-27: Reservation of protected memory



IOMEM

The `iomem` directive in the guest configuration file maps the pages corresponding to protected memory to each guest address map. The code example shows the use of `iomem`:

```
iomem= [ "IOMEM_START, NUM_PAGES[@GFN]"

Where:
IOMEM_START    =    The physical page number corresponding to the start
address
```

```

NUM_PAGES      =      Number of physical pages from IOMEM_START to share
GFN            =      The guest frame number where the mapping
will start in the guest's address space

```

Example:

In the XEN guest 1 configuration file (guest1.cfg) we add the following line:

```
iomem = [ '0xC0000,0x8000@0x12000' ]
```

Which gives:

0xC0000 as the physical start address of the protected memory as defined in the DOM0 device tree, and

0x8000 makes the first 128MB of protected memory visible from within Guest#1.

For each other guest the IOMEM_START (0xC0000) should be adjusted in the guest configuration file (guestN.cfg)

Guest Device tree

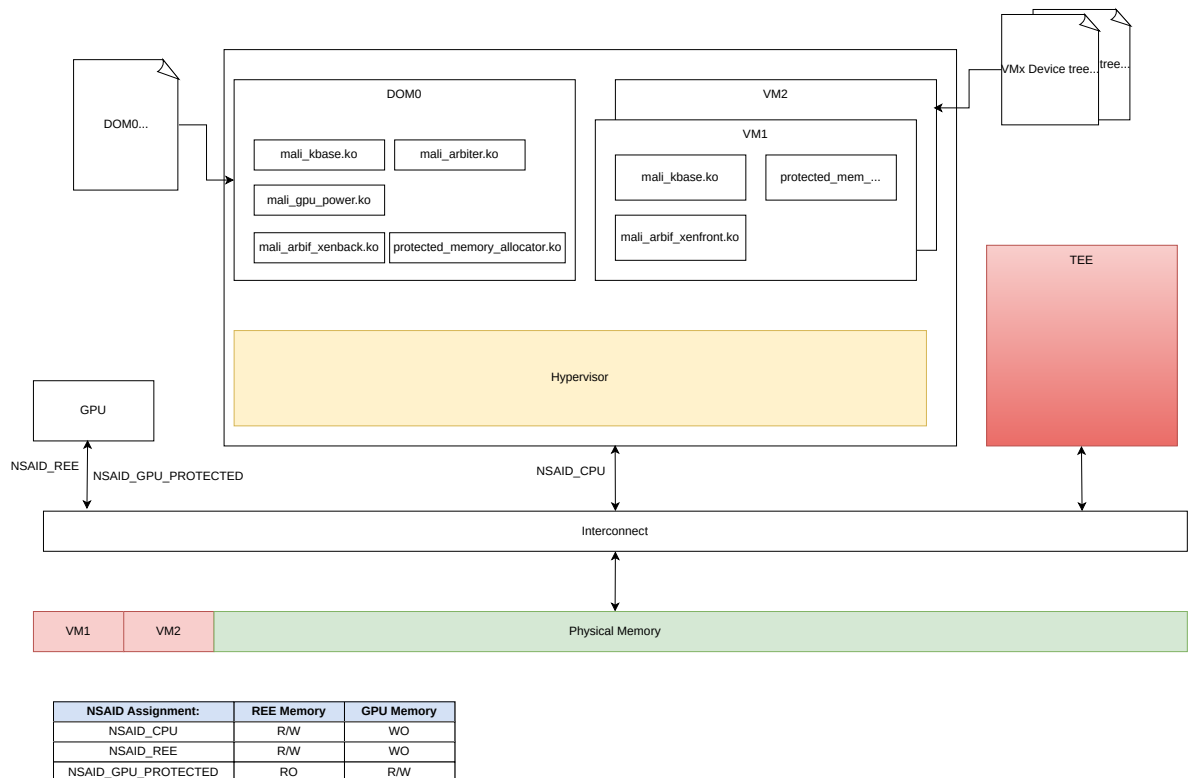
In the partial Device tree for all guests, add the reserved memory region definition. The address in the guest address space must not conflict with any other guest. The example code shows how to add the memory:

```

reserved-memory {
    #address-cells = <2>;
    #size-cells = <2>;
    ranges;

    mali_protected: mali_protected@12000000{
        compatible = "mali-reserved";
        reg = <0x0 0x12000000 0x0 0x80000000>;
    };
};

```


Figure 2-28: Example PV System Implementing TZMP1

- TZMP1 with PV software architecture -

2.13 Software security considerations for virtualization

Virtualized systems can provide secure operation. The level of security depends on several aspects of the implementation.

You must consider several security issues and suggested mitigations when you integrate a system. To ensure that the virtualized system gives the wanted level of security, address these issues:

Separation between virtual machines

Virtual machines (VMs) must be isolated from each other. VMs must not access the address space of other VMs. Protection must be guaranteed for:

- Memory access from the CPU.
- Other peripherals accessing the memory, such as the GPU.

You can implement separation between VMs using:

- A second stage *Memory Management Unit* (MMU) and a system MMU, owned and configured by the hypervisor,
- A *Memory Protection Unit* (MPU) owned and controlled by the hypervisor.

An important implication of this security requirement is that the hypervisor controlling the S2-MMU or the MPU must be:

- Trusted.
- Run in a higher-level execution level (for example EL2).

Quality of Service and freedom of interference between VMs

The current reference implementation of the paravirtualization solution provides a basic *Quality of Service* (QoS) guarantee. There might be occasions when the arbiter asks a guest VM to yield the GPU and the VM does not yield in time. Failure to yield in time impacts other VMs sharing GPU. The reference arbiter implementation provides a mechanism to force a virtual machine to yield the GPU after a configurable amount of time. See [2.1.1 Arbiter](#) on page 16.

The implication of this security requirement is that the arbiter must implement fair scheduling of the GPU between VMs. Therefore you must choose the various timeouts, that the arbiter uses to control the yield behavior, in accordance with the required scheduling policy.

The arbiter must be trusted.

GPU HARD_STOP

Some implementations support HARD_RESET by the VM. When granted the GPU, a guest VM has full access to the GPU registers and can then issue a HARD_RESET command. This process might leave the system in an undefined state. To mitigate this situation, you must ensure that the GPU driver cannot issue the HARD_RESET from a guest VM.

Instrumentation and debug interfaces

Debugging and instrumentation tools are generally reserved for development. Information disclosed through these tools can be exploited in a side-channel attack. For example, instrumentation data might disclose information about a memory space of a process. Malicious use of this information might take the form of spoofing, tampering or a *Denial of Service* (DoS) attack. These mitigations are recommended:

- Disable instrumentation in production systems. For example, disable the mapped utility.
- Restrict dmesg access to root.
- Do not relax the default root-only access to the debugfs output.
- Do not log error conditions related to protected mode.

Protected content management

The safety island or the secure firmware sets the STREAM_IDs or NSAIDs for the GPU. Incorrect implementation of STREAM_ID assignments and MMU/MPU configuration could make the protected memory region of a VM accessible to another VM via the GPU. Memory transactions from each virtual machine are therefore uniquely identified. There must be enforcement of the required memory protection policy. The system enforces this policy through:

- The safety island configuration of the MPU.
- The domain hypervisor control of a stage 2 MMU.

Therefore an MPU configured by the safety island or a stage 2 MMU controlled by the domain hypervisor must enforce the required memory protection policy.

Sysfs interfaces

Some sysfs entries allow interaction with the arbiter and allow GPU configurations to be set (for example, partition configurations). It is important to ensure that only a privileged process (for example root privileges) accesses such entries. Also, make sure that the interface is disabled for systems not using sysfs.



When you replace the sysfs interface with your own implementation, make sure that your design has minimal impact on the arbiter performance.

Interfaces with power module

The reference implementation does not implement any sort of authentication between arbiters and the power module. To avoid any kind of spoofing attack, apply one of these measures:

- Add an authentication mechanism to the communication protocol.
- Make all the arbiters in the system trusted.

Safety island isolation

Security for inter-domain isolation, inter-VM isolation, and memory protection depends on mitigating various vulnerabilities:

Potential vulnerabilities

- A non-privileged domain tries to access memory pages (System, Assign) owned by the safety island.
- A domain tries to access another domain owned registers pages (Resource Group, Partition Control and Configuration):
 - From the same bus.
 - From another bus.
- A domain tries a DoS attack to prevent another domain from accessing its assigned registers pages (Resource Group, Partition Control and Configuration):
 - From the same bus.
 - From another bus.

Mitigation

- You must use a stage-3 MMU or MPU to isolate the domains (hypervisors) from each other and from the safety island.
- The safety island must control the stage-3-MMU or MPU.

Information disclosure through GPU yield frequency

It is theoretically possible that kbase can determine how active another VM is, based on the frequency of GPU yields and the GPU operating frequency. The frequency can be determined either from the arbiter or by calculating the execution rate on the GPU.

There is a restriction on this information through combinations of:

- The granularity at which VMs are time-sliced
- The amount the GPU frequency is altered.

A very high frequency of time-slicing or frequency change is required to infer specific behavior of another VM.

Information disclosure through GPU sharing

The GPU can contain, or have access to, restricted information belonging to a VM. To prevent this information being available to another VM, the reference arbiter resets the GPU and reconfigures the SMMU before passing the GPU on. Any alternate custom implementation of the arbiter must uphold this principle to maintain cross-VM security.

Tampering through system configuration data

An attacker or unprivileged process can make changes in the system configuration through actions such as:

- STREAM_IDs assignment
- Resource group to bus
- Slices to resource groups
- Access windows to resource groups

The system configuration data must have adequate protection against these actions.

3. Message protocol

The message protocol defines the communication messages passed between the arbiter and *Virtual Machines* (VMs) to negotiate access to GPU resources.

The communication between the arbiter and the *Virtual Machines* (VMs) is organized into four main blocks. These blocks are data passing, protocol management, message handling, and communication interface for API calls.

Data passing

This block manages the passing of actual data, for example bits encoding message, between the arbiter and multiple VMs through a physical or virtual channel.

Protocol management

Message handlers use this block to encode messages to be sent and decode the received messages. Protocol management is a data passing mechanism.

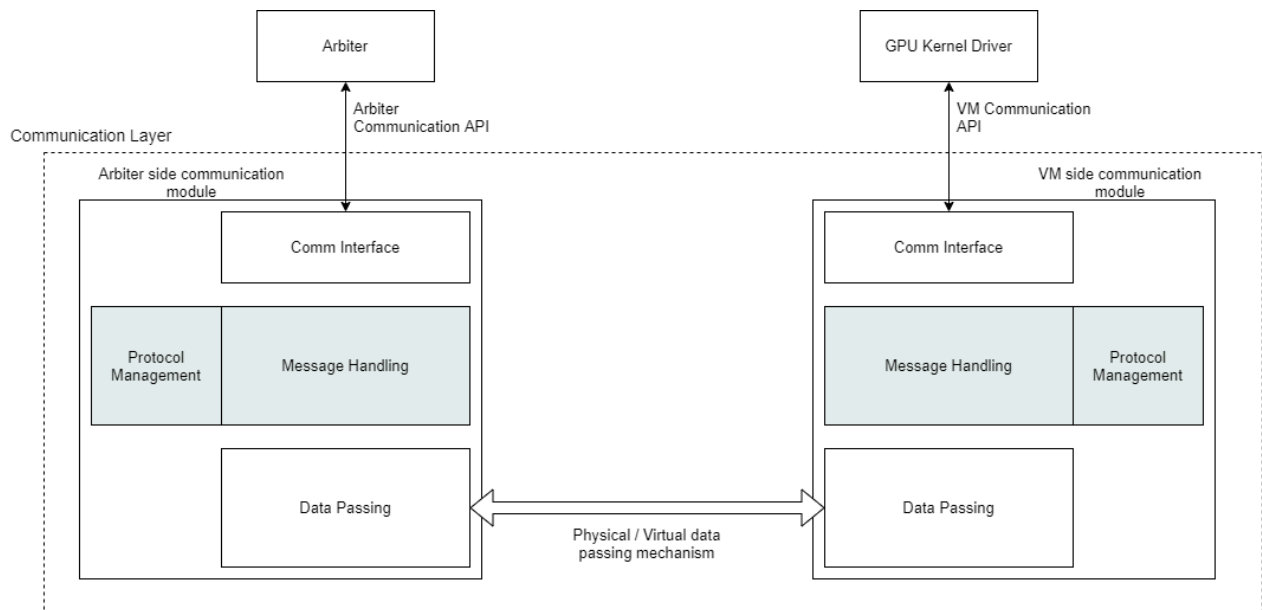
Message handling

This block handles the protocol processes, for example the initialization process, the restarting or recovering process, or yielding. Those processes provide an abstraction layer around the implementation.

Communication interface for API calls

This block is the C interface used by kernel modules.

Figure 3-1: Arbiter VM communication layer



The reference stack implements communication interfaces for the arbiter and the VM over a message passing protocol, shown in the arbiter VM communication layer figure. This messaging

protocol is well defined and is backwards compatible. Although it is possible to re-implement the interfaces without adhering to the message passing protocol, Arm® recommends that the protocol is used. We do not provide any guarantee on interface backward compatibility. Where possible, however, we intend to keep the protocol forward and backward compatible with future releases.

The message protocol is designed with the following characteristics:

Flexibility

Support for single or multiple VMs, or domains, up to the maximum number.

Support for the arbiter and GPU kernel driver instances in the same VM, with no assumption about module load ordering.

Support for both built-in and loadable kernel modules.

Maintainability and portability

The same message interface is used for both paravirtualization and hardware virtualization.

The messaging protocol uses a simple 64-bit message format with reserved bit ranges in the payload for future use.

Robustness

Handshaking between VMs and the arbiter:

- Enables recovery from restarts of the arbiter domain, a VM, or both.
- Allows a VM to return to a known stable state after a restart, regardless of the state of other VMs, or other domains.
- Avoids losing ongoing GPU work on VMs when the arbiter is reset.
- Enables arbitrary initialization order of arbiter and VMs.

Backward compatibility

The protocol definition allows arbiters and VMs to negotiate the highest commonly supported protocol version, which enables protocol backward compatibility

Performance

To reduce boot time, messages have been defined in a way that minimizes the number of round trips between the arbiter and a VM during initialization.

3.1 Protocol versioning and backward compatibility support

It is possible to have a system which includes VMs that use different versions of the reference software but continue to cooperate with each other by using the arbiter. For example, you have a VM with an older version of the DDK which you cannot easily update, but you can make updates to the arbiter and the guest VMs to let them all work together.

A protocol version message consists of 2 parts:

Minimum supported version

Any version earlier than this version cannot be supported.

Current or maximum supported version

Any version later than this version cannot be supported.

A protocol version consists of a single number that is incremented every time when the protocol is changed.

When initializing communication between the arbiter and a VM, the two sides must agree on a protocol version. This agreement is referred to as the protocol handshake and uses the `ARB_VM_INIT` and `VM_ARB_INIT` messages. The handshake initiator is an arbiter or a VM, and it sends the maximum version that can be a current version the arbiter or the VM supports. The responder is a VM or an arbiter that acknowledges the handshake and responds with the version it uses. That version to use will be the minimum of the maximum versions reported.



There is no error handling in the reference messaging protocol if the protocol versions do not overlap.

In the reference implementation of our modules, when there is no overlap of supported versions between the communication modules, the communication is not allowed between those modules and an error message is printed.

3.2 Protocol message format

Each message has an ID, which defines the type of message, and a payload. The payload definition depends on the type of message, and some messages do not use the payload at all.

Message format

Table 3-1: Message format

Field	Size (bits)	Value
Message ID	8	VM_ARB_INIT VM_ARB_GPU_IDLE VM_ARB_GPU_ACTIVE VM_ARB_GPU_REQUEST VM_ARB_GPU_STOPPED ARB_VM_INIT ARB_VM_GPU_STOP ARB_VM_GPU_GRANTED ARB_VM_GPU_LOST
Payload	56	Message dependent

3.2.1 Messages from arbiter to VM

Message from arbiter to VM

The `ARB_VM_INIT` message is used to start or complete an initialization handshake.

Table 3-2: ARB_VM_INIT

Field	Size (bits)	Value
Message ID	8	<code>ARB_VM_INIT</code>
Ack	1	1: If ACK is sent by <code>VM_ARB_INIT</code> . 0: If ACK is not sent by <code>VM_ARB_INIT</code> .
Version	7	If Ack = 0, the value returned is the Max version supported. Otherwise, the value returned is the protocol version.
(Field reserved)	8	-
Max L2 slices	8	0: If using paravirtualization
Max core mask	32	0: If using paravirtualization

`ARB_VM_INIT` can either be used to start the initialization sequence if the Ack value equals 0, or to complete the initialization sequence if the Ack value equals 1. In normal cases, it is expected that the arbiter starts the initialization to minimize latency. However, if a guest *Virtual Machine* (VM) restarts, it must restart the initialization.

The max L2 slices and the max core mask fields are used with hardware virtualization to allow the guest to allocate sufficient resources for the maximum-sized partition. For paravirtualization, `ARB_VM_INIT` is not required, and the arbiter has no knowledge of the values. In those cases, 0 is used.

ARB_VM_GPU_STOP

The `ARB_VM_GPU_STOP` message is used to inform the VM that the process of yielding the GPU must start.

When receiving this message, the VM driver must ensure that any existing executing work is preempted based on the available preemption granule. The driver must put itself into a state where the GPU is effectively considered unpowered. The VM must respond with a `VM_ARM_GPU_STOPPED` when the GPU is idle.

The arbiter issues the `mali_ARB_GPU_STOP_REQUESTED` ftrace event when handling the `ARB_VM_GPU_STOP` message.

Table 3-3: ARB_VM_GPU_STOP

Field	Size (bits)	Value
Message ID	8	<code>ARB_VM_GPU_STOP</code>
(Field reserved)	56	-

ARB_VM_GPU_GRANTED

The `ARB_VM_GPU_GRANTED` message is used to inform the VM that it can start scheduling work on the GPU immediately. The message is sent again when the frequency changes.

The arbiter issues the `mali_ARB_GPU_GRANTED` ftrace event when handling the `ARB_VM_GPU_GRANTED` message.

Table 3-4: ARB_VM_GPU_GRANTED

Field	Size (bits)	Values
Message ID	8	<code>ARB_VM_GPU_GRANTED</code>
Frequency	24	In kHz 0kHz, if there is no power management.
(Field reserved)	32	-

Frequency is required when the arbiter implements *Dynamic Voltage and Frequency Scaling* (DVFS), so that the frequency can be included in the instrumentation.

Uses of the frequency do not accumulate errors, so kHz granularity is sufficient.

The Mali™-G78AE GPU is expected to support up to 850MHz, so 24 bits (max of 16GHz) is sufficient.

ARB_VM_GPU_LOST

This `ARB_VM_GPU_LOST` message is used to inform the VM that the access to the GPU has been forcibly removed.

If the GPU has been in use when this message is received, the VM GPU kernel driver must take the appropriate steps to ensure that the operations still pending a response from the hardware are thrown away, and the resulting failure propagated appropriately.

The arbiter issues the `mali_ARB_GPU_LOST` ftrace event when handling the `ARB_VM_GPU_LOST` message.

Table 3-5: ARB_VM_GPU_LOST

Field	Size (bits)	Value
Message ID	8	<code>ARB_VM_GPU_LOST</code>
(Field reserved)	56	-

3.2.2 Messages from VM to arbiter

VM_ARB_INIT

The `VM_ARB_INIT` message is used to start or complete an initialization handshake.

Table 3-6: VM_ARB_INIT

Field	Size (bits)	Value
Message ID	8	<code>VM_ARB_INIT</code>
Ack	1	1: If ACK is sent by <code>ARB_VM_INIT</code> . 0: If ACK is not sent by <code>ARB_VM_INIT</code> .
Version	7	If Ack = 0, the value returned is the Max version supported. Otherwise, the value returned is the protocol version.
Request	1	1: for requesting the GPU. 0: for not requesting the GPU.
(Field reserved)	47	-

`VM_ARB_INIT` can either be used to start the initialization sequence if the Ack value equals 0, or to complete the initialization sequence if the Ack value equals 1. In normal cases, it is expected that the arbiter starts the initialization to minimize latency. However, if a guest *Virtual Machine* (VM) restarts, it must restart the initialization.

The max L2 slices and the max core mask fields are used with hardware virtualization to allow the guest to allocate sufficient resources for the maximum-sized partition. For paravirtualization, `VM_ARB_INIT` is not required, and the arbiter has no knowledge of the values. In those cases, 0 is used.

VM_ARB_GPU_IDLE

The `VM_ARB_GPU_IDLE` message is used to inform the arbiter about GPU idleness within the VM. The arbiter uses this message to make scheduling and power decisions. For example, a VM reports that the GPU has gone idle and the message can be used as a scheduling point to allow another VM to be scheduled based on the usage of the GPU.

`VM_ARB_GPU_IDLE`, together with `VM_ARB_GPU_ACTIVE` might be used to calculate GPU utilization for *Dynamic Voltage and Frequency Scaling* (DVFS), although the calculation source code is handled differently in the reference arbiter.

Table 3-7: VM_ARB_GPU_IDLE

Field	Size (bits)	Value
Message ID	8	<code>VM_ARB_GPU_IDLE</code>
(Field reserved)	56	-

VM_ARB_GPU_ACTIVE

The `VM_ARB_GPU_ACTIVE` message is used to inform the arbiter of GPU activity within the VM, and the Arbiter can also use the message to make scheduling decisions.

kbase issues the `mali_ARB_GPU_STARTED` ftrace event when handling the `VM_ARB_GPU_ACTIVE` message.

Table 3-8: VM_ARB_GPU_ACTIVE

Field	Size (bits)	Value
Message ID	8	<code>VM_ARB_GPU_ACTIVE</code>
(Field reserved)	56	-

VM_ARB_GPU_REQUEST

This `VM_ARB_GPU_REQUEST` message is used to inform the arbiter that a VM has work pending for the GPU. Once the request is issued, the VM has to wait until the it is granted ownership via a `ARB_VM_GPU_GRANTED` message which is being received in the VM from the arbiter.

kbase issues the `mali_ARB_GPU_REQUESTED` ftrace event when handling the `VM_ARB_GPU_REQUEST` message.

Table 3-9: VM_ARB_GPU_REQUEST

Field	Size (bits)	Value
Message ID	8	<code>VM_ARB_GPU_REQUEST</code>
(Field reserved)	56	-

VM_ARB_GPU_STOPPED

This `VM_ARB_GPU_STOPPED` message is used to inform the arbiter that the VM has yielded the GPU and that GPU is able to be reassigned to another VM.

kbase issues the `mali_ARB_GPU_STOPPED` ftrace event when handling the `VM_ARB_GPU_STOPPED` message.

Table 3-10: VM_ARB_GPU_STOPPED

Field	Size (bits)	Value
Message ID	8	<code>VM_ARB_GPU_STOPPED</code>
Request	1	1: for requesting the GPU. 0: for not requesting the GPU.
(Field reserved)	55	-

3.3 Messaging protocol processes with example message sequences

3.3.1 Initialization message protocol

The communication between arbiter and VM is initialized when both agree on the latest possible version supported. This agreement is called the protocol handshake.

Under normal circumstances, at startup, the arbiter initiates the handshake with each of its VMs, although it is also possible for a VM to initiate the handshake if the VM restarts.

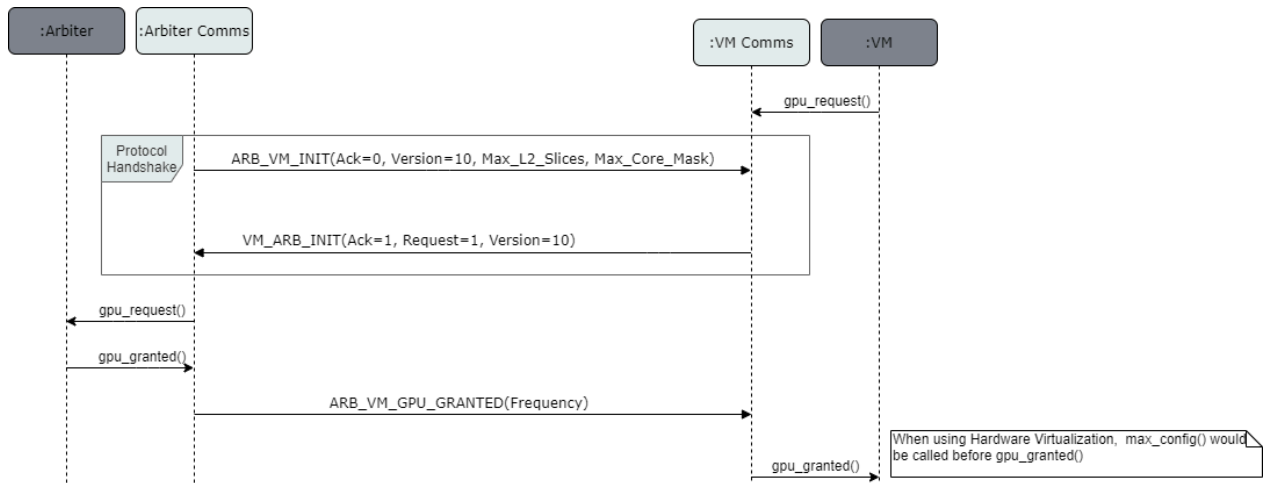
3.3.1.1 Typical initialization

A typical protocol handshake is initiated by the arbiter with each VM.

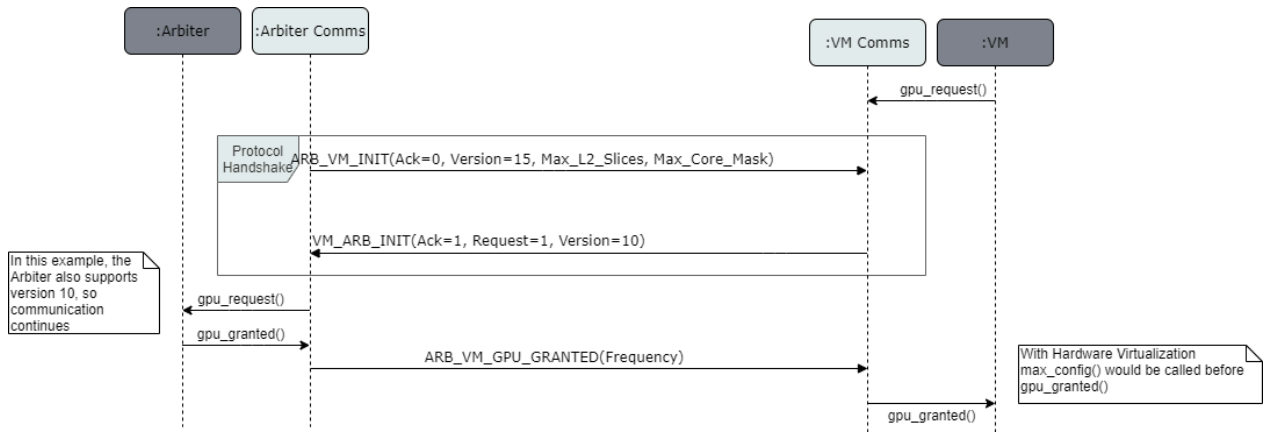
During the initialization, the arbiter and a VM agree on a protocol version. The arbiter also informs the VM of its maximum configuration to allow the VM to allocate sufficient resources. In these circumstances, when it responds to the handshake, the VM requests access to the GPU.

Matching version ranges

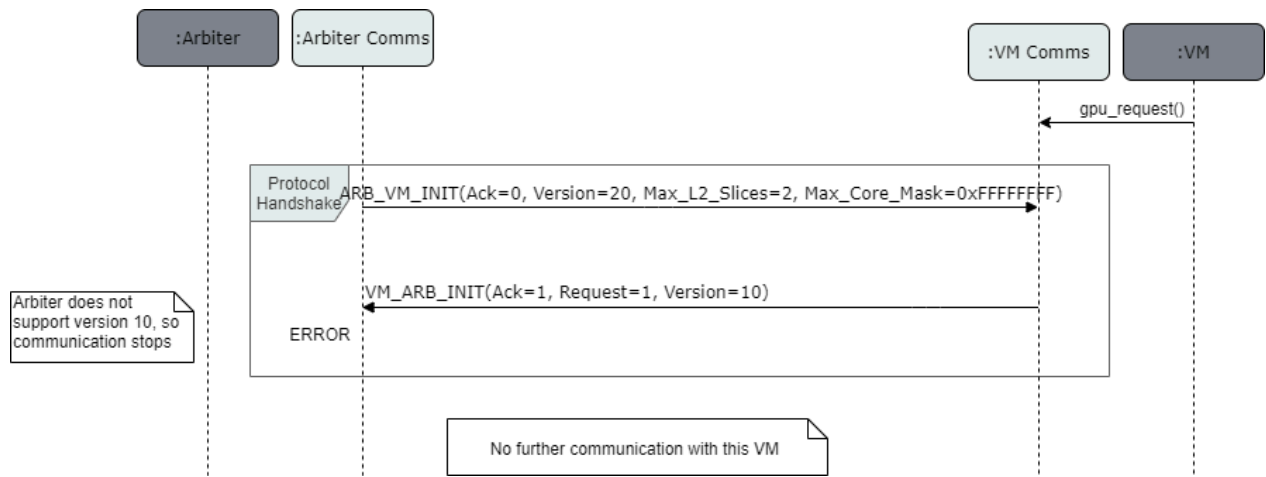
In the example of a normal initialization, the arbiter and the VM have matching version ranges, for example both support a maximum of version 10.

Figure 3-2: Normal initialization**Version matched when the arbiter version is newer than VM**

In the example of version matched, the arbiter domain version is newer than the VM version, and their versions can match.

Figure 3-3: Version matched**Version unmatched when the arbiter version is newer than VM**

In this example of version unmatched, the arbiter domain version is newer than the VM version, and the arbiter does not support the VM version, so there is no match.

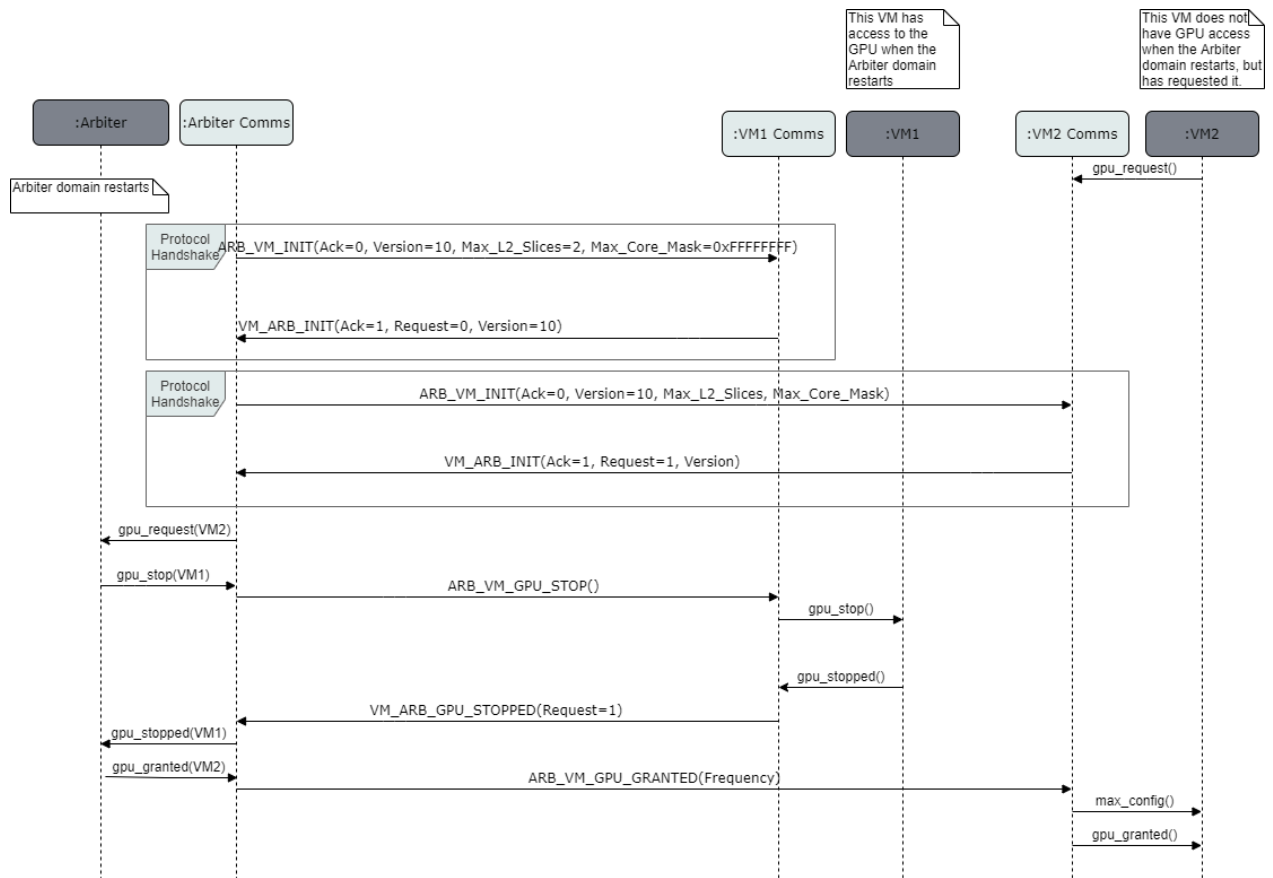
Figure 3-4: Version unmatched

3.3.1.2 Initialization after arbiter restart

After the arbiter domain is restarted, the arbiter initiates the protocol handshake with each of the VMs.

In hardware virtualization, the arbiter initiates the yield sequence with any of its enabled VMs to synchronize the communication with the VM again.

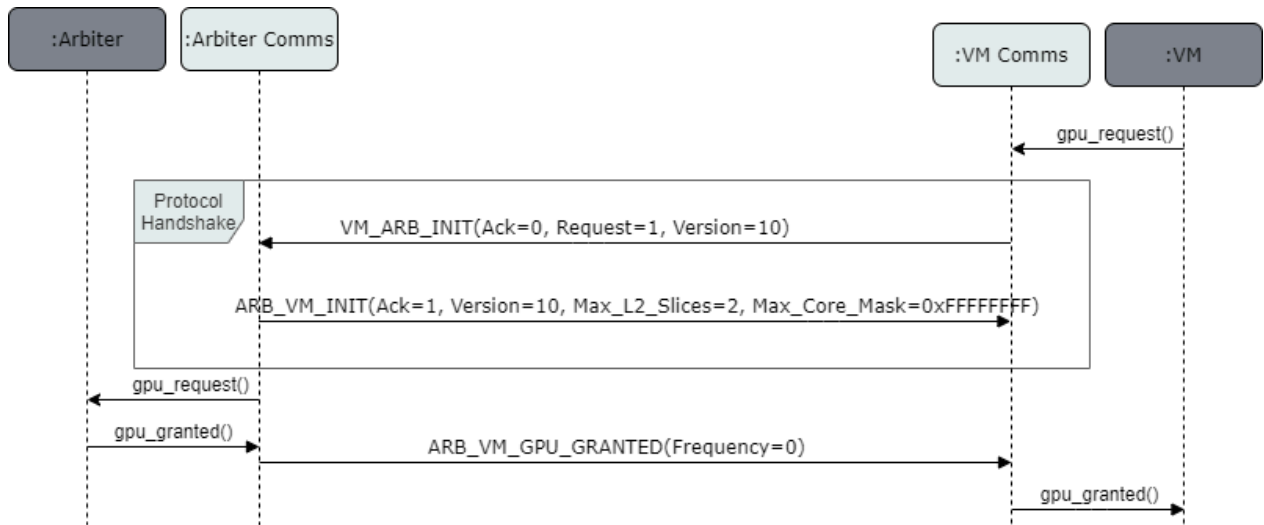
This figure shows the arbiter initiating the yield sequence in hardware virtualization.

Figure 3-5: Arbiter restart sequence

3.3.1.3 Initialization after VM restart

When a VM domain is restarted, the VM initiates the protocol handshake and requests the GPU.

This figure shows a VM initiating the handshake.

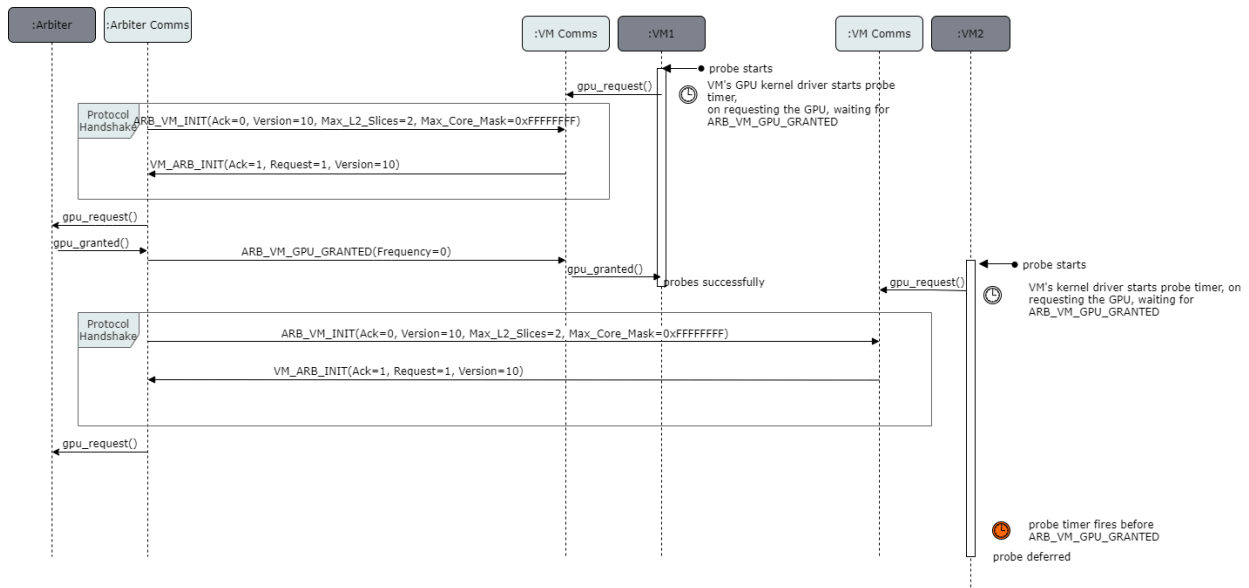
Figure 3-6: A VM initiates handshake

3.3.1.4 GPU kernel driver deferred probe on delayed GPU granted

Virtual Machines (VMs) may be started in parallel to minimize boot time.

If a VM is ready before the arbiter or if a VM cannot get the GPU in time during the initial driver probe phase, like the following figure of deferred probe, the driver probe will be deferred after a timeout.

If the VM does not receive the `ARB_VM_GPU_GRANTED` message within the defined timeout, the probe is deferred. The timeout starts when the VM requests the GPU during probe, and the timeout stops when the `ARB_VM_GPU_GRANTED` is received.

Figure 3-7: Deferred probe

The value of the probe timeout is configurable using a GPU kernel driver module parameter, `gpu_req_timeout`, to the GPU kernel driver. The default value is 1ms, but you must change it depending on the system configuration. For example, where there is more than one VM requesting access to the GPU at startup, this timeout must be extended to allow each VM enough time to access the GPU and complete its initialization.

There is a reprobe mechanism implemented for both paravirtualized and hardware virtualized cases in the reference software. For hardware virtualization, the reprobe happens when the `ARB_VM_GPU_GRANTED` arrives, and for paravirtualization the reprobe happens when the virtual data passing channel is established.

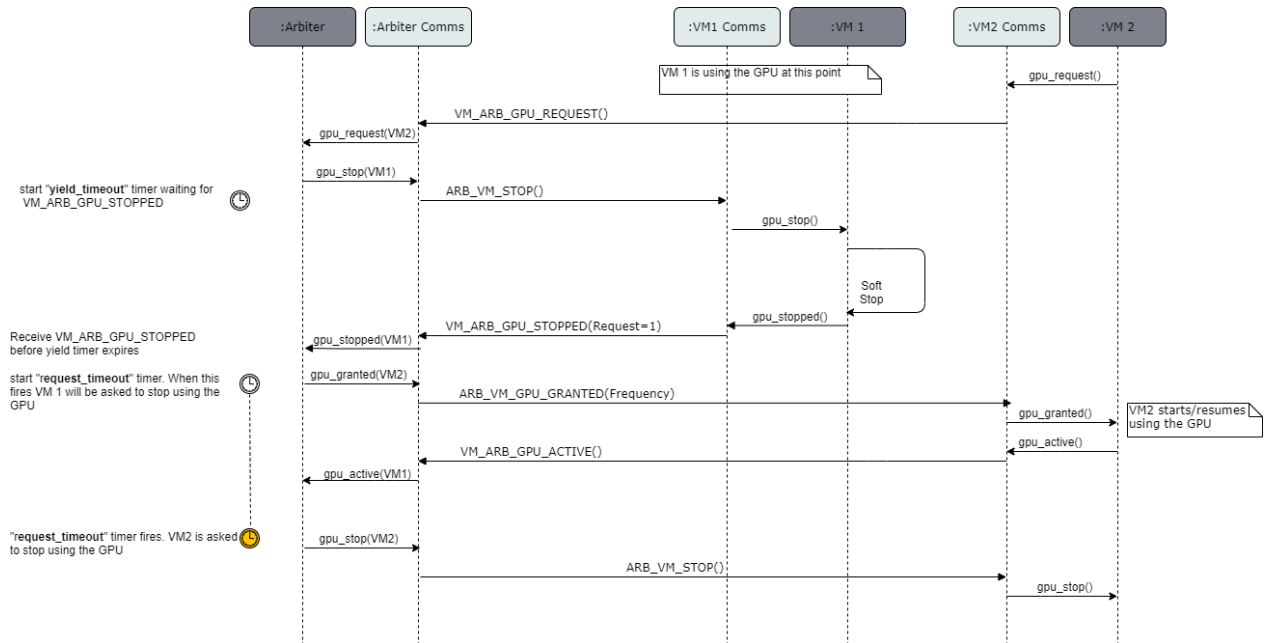
These examples are where deferring probe is necessary:

- Latency of the arbiter due to other VMs using the GPU (normal time slicing latency).
- Atypical latencies of switching times of other VMs.
- Wrong configuration of the `gpu_req_timeout` that is not set with appropriate consideration for the normal system latencies and other VMs running.
- Latency of communication setup, for example the delayed arbiter boot or arbiter restart.
- Delayed or missing VM setup in the hardware virtualization. The *Access Window (AW)* is not assigned to any partition via `sysfs`.

3.3.2 GPU yield

The sequence of arbitration events in the following figure of GPU yields shows where the VM yields the GPU in time, because the soft-stop time is sufficiently short.

Figure 3-8: GPU yields



```
request_timeout
```

This measurement refers to the GPU time, in ms, given to a VM before switching to another VM when multiple VMs have requested the GPU.

```
yield timeout
```

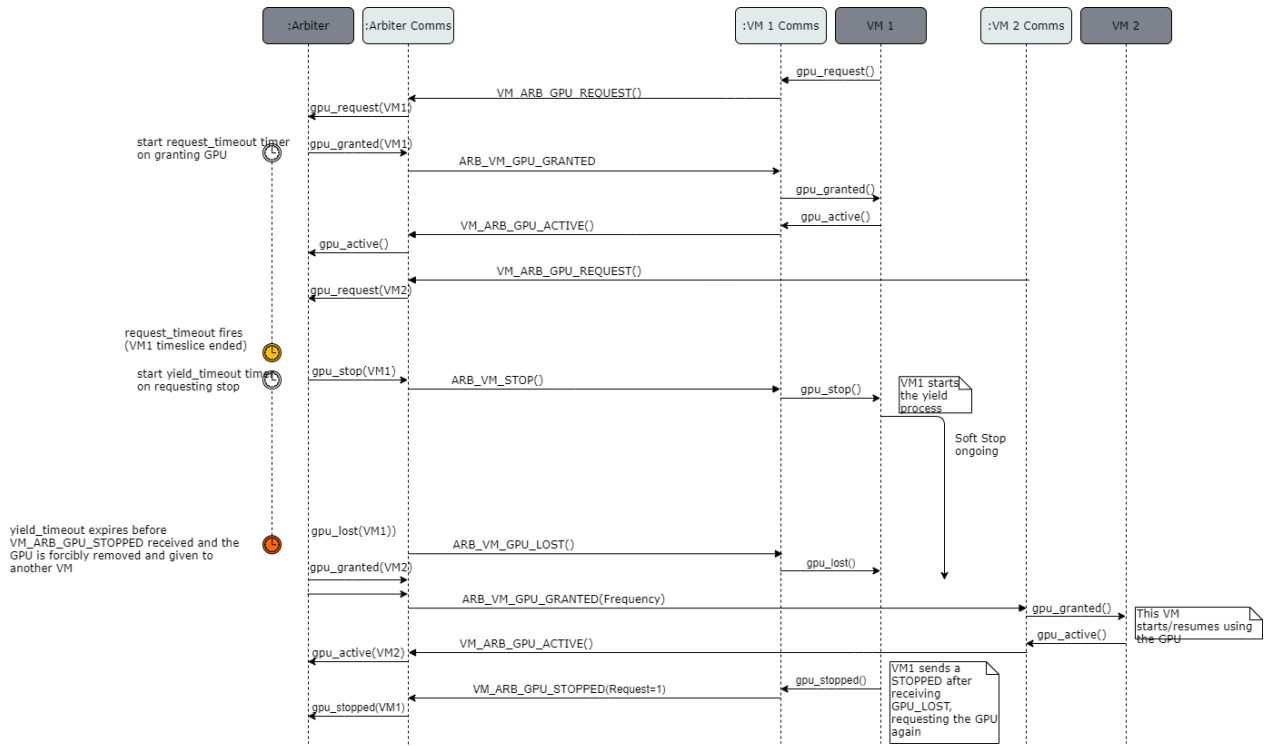
This measurement refers to the maximum time, in ms, given for the VM to stop before forcibly removing the GPU.

request timeout and yield timeout are configurable as arbiter module parameters.

3.3.3 GPU lost

The sequence of arbitration events illustrated in the following figure of GPU lost shows the soft-stop taking too long and the arbiter forcibly removing the GPU from VM 1.

Figure 3-9: GPU lost



On receiving a `GPU_LOST`, a VM must reset its internal state, and let the user-side be aware that the GPU work has not been completed, so that the application using the GPU can recover.

VM 1 must then request the GPU again, either by setting the request flag in a `VM_ARB_GPU_STOPPED` message, or by sending a `VM_ARB_GPU_REQUEST` message.

Appendix A Limitations on virtualization in GPUs

The use of paravirtualization in systems with GPUs has some limitations. The limitations depend on the system configuration and the way the system responds to misbehaving modules.

In the reference solution, there are several limitations:

- The GPU cannot preempt shaders. Therefore some content takes too long to soft stop and is unsuitable for paravirtualization without changes.
- There is limited Quality of Service for virtual machines. The virtual machines do affect each other. It is not possible to guarantee low boundaries to the number of frames per second.
- The virtual machines are aware that they are virtualized and need modifying.
- Arbiter takes cycles to run.
- Power management is limited.
- Because of the yield time, there is an effective upper limit to the number of supported virtual machines.

The performance of cooperative sharing of the GPU is subject to (but not limited to) these limitations:

- Misbehaving drivers might hold spinlocks or run at the interrupt (IRQ) level for too long. These drivers might impact the latency of messaging between the arbiter and kbase. The latency depends on the communication system adopted by platform.
- Misbehaving platform drivers might tie up *Central Processing Unit* (CPU) cores for too long (by running KThreads without yielding). These drivers might also impact switching performance when the available CPU cores are limited in either host or guests.
- System Memory Management Unit or Memory Protection Unit reconfiguration complexity.
- I/O Memory Management Unit reconfiguration of GPU registers.
- Power management costs:
 - Powerdown and powerup of GPU, level two memory, and CORES.
- GPU reset time. The arbiter resets the GPU every time it assigns a new virtual machine.
- Content yield time, that is `SOFT_STOP`.
 - Switching virtual machines only works if the virtual machines can yield fast enough. For example, if two virtual machines need to display content at 60FPS, then each virtual machine must render its content for the frame and yield each 1/60sec. Yield time involves several activities:
 - A virtual machine requests the GPU from the hypervisor.
 - The hypervisor asks the current virtual machine to stop.

- The current virtual machine soft-stops running jobs and notifies hypervisor when it is done. If the jobs are long-running, soft-stops can take a significant time (for example complex shaders).
- The hypervisor resets the GPU and informs the waiting virtual machine on completion.
- The next virtual machine resumes with the work it had scheduled.
- The most significant time is expected to be the time taken to soft-stop the GPU.
- This yield time depends on content and is outside the control of the DDK and the GPU.

Appendix B Xen hypervisor

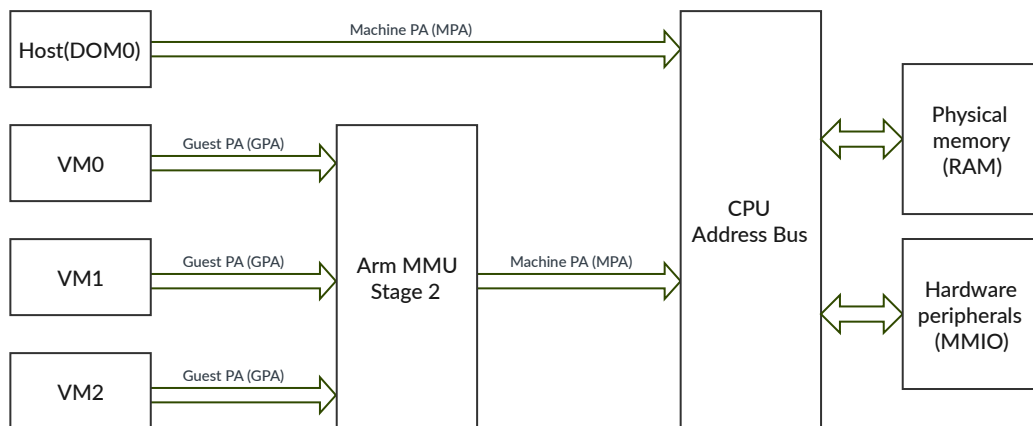
This appendix covers examples of using the paravirtualization and hardware virtualization stacks with the Xen Hypervisor.

Xen is a type-1 hypervisor which runs directly on hardware and everything else in the system runs as a VM on top of Xen.

The following diagrams show how the Xen hypervisor is used in GPU paravirtualization.

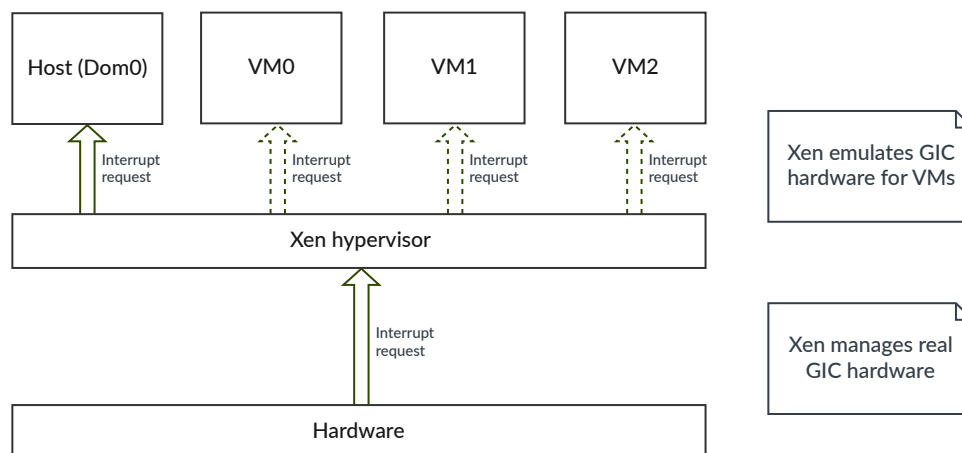
Xen memory mapping

Figure B-1: Xen memory mapping



Xen interrupt management

Figure B-2: Xen interrupt management



Device tree entries of the Xen hypervisor

The GPU device tree node (in the HOST domain)

```
gpu: mali@E82C0000 {
```

```

status = "disabled";
xen_passthrough = <1>;
reg = <0x0 0x11000000 0x0 0x4000>;
interrupts = <0 258 4 0 259 4 0 260 4>;
interrupt-names = "JOB", "MMU", "GPU";
hwreg = <0x0 0xE82C0000>;
...
};

```

B.1 Paravirtualization worked example

This example details the paravirtualization specific components, and Xen configuration necessary to use the stack.

Reference components

- **mali_arbif_xenback**

The `mali_arbif_xenback` component provides reference source code for a Xen backend driver. The component connects to the arbiter and provides communication between VMs and the frontend.

mali_arbif_xenfront

The `mali_arbif_xenfront` component provides reference source code for a Xen frontend driver. The component connects to `xbase` and provides communication between VMs and the backend.

host-xen.patch

The `host-xen.patch` component is a patch that adds Mali PV support.

Xen backend driver

When the reference `mali_arbif_xenback.ko` module is loaded, it searches for an arbiter device in the device tree. If an arbiter is found, then the Xen backend driver calls:

arb_reg_assign_if

The `arb_reg_assign_if` driver call registers the VM assign interface with the arbiter, providing a callback function for assigning the GPU to a VM. The Xen backend driver reference implements this callback function by making a hypercall into Xen.

register_vm

The `register_vm` driver call registers communication between the VM functionality and the arbiter. The call enables the arbiter to send and receive messages with the Xen frontend driver.

Xen frontend driver

When the reference `mali_arbif_xenfront.ko` module is loaded, it attempts to connect with the Xen backend driver. For this connection to work, the two drivers must be linked in the Xen store. This linking can be done by making the following `xenstore` writes:

```

/local/domain/<DomFE>/device/mali_arbif_xen/0/backend-id <DomBE>
/local/domain/<DomFE>/device/mali_arbif_xen/0/backend
/local/domain/<DomBE>/backend/mali_arbif_xen/<DomFE>/0
/local/domain/<DomBE>/backend/mali_arbif_xen/<DomFE>/0/frontend-id <DomFE>
/local/domain/<DomBE>/backend/mali_arbif_xen/<DomFE>/0/frontend

```

```
/local/domain/<DomFE>/device/mali_arbif_xen/0
/local/domain/<DomFE>/device/mali_arbif_xen/0/state 1
/local/domain/<DomBE>/backend/mali_arbif_xen/<DomFE>/0/state 1
```

Where `<DomFE>` is the frontend domain ID and `<DomBE>` is that backend domain ID. The `xenstore` writes can be done before or after loading the modules.

The Xen frontend driver also creates an `ARBIF` platform device from the `ARBIF` node in the device tree. `kbase` attempts to connect to this driver during probe, if the `arbiter_if` attribute points to the `ARBIF` device.

Changes that are needed in Xen hypervisor support

The Xen patch `host-xen.patch` file includes:

- Support for GPU timeslicing mode. When timeslicing mode is enabled, the hypervisor expects to find an active GPU node in the device tree.
- A new Xen hypercall called `arb_gpu_op`. This hypercall provides a `ARB_HYP_ASSIGN_VM_GPU` command for assigning the GPU to a VM. When the Xen hypervisor boots, the GPU defaults to being unassigned. This hypercall is only valid when GPU timeslicing mode is enabled.
- SMMUv3 support
- Fixes to allow Xen to reroute interrupts while guests are running.
- MPU support so that guest physical memory can be partitioned into separate predefined contiguous regions.

Assigning the GPU to a domain

The Xen hypercall for assigning the GPU to a VM `ARB_HYP_ASSIGN_VM_GPU` is implemented in the following way:

1. If there is a virtual machine currently assigned the GPU and it does not match the one being assigned, then it must be unmapped and the GPU reset:
 - a. Unmap the GPU registers from the current VM. The register memory must be replaced with hypervisor managed memory-mapped I/O so that reads always return 0 and writes are ignored.
 - b. Disable and clear the GPU Mali interrupts.
 - c. Unbind interrupts (IRQs) from the VM
 - d. Disable the interrupts (IRQs) inside the Global Interrupt Controller clear any interrupts that are already queued.
 - e. Reset the GPU.
2. If the VM to be assigned does not already have the GPU and has a valid ID (not `ARB_HYP_INVALID_DOMAIN_ID`), then:
 - a. Reconfigure the system Memory Management Unit for the new VM.
 - b. Map the GPU registers to the new VM.
 - c. Reroute the interrupts (IRQs) to the new VM and enable them in the Global Interrupt Controller.
3. In the special case that the VM is already assigned the GPU, only a GPU reset is required.

B.2 Hardware virtualization worked example

This example can help users who are already familiar with Xen to extend an already working system to support running the Mali™ GPU virtualization stack.

The Xen Hypervisor platform setup is out of scope of the Mali™ GPU documentation and you must use it in conjunction with the relevant [documentation and references](#).

B.2.1 Example System Deployment

In the register map *Physical Address* (PA) range, the GPU exposes multiple access windows and the registers to control them. The system integrator decides how these are mapped into DOM0 and DOMU VMs.

For this example, we put the privileged functions in DOM0:

- System
- Assign
- Arbiter
- Resource Group
- Partition Configuration
- Partition Control

The system and assign functions have higher level privileges, while arbiter, resource group, partition configuration, and partition control have resource group level privileges.

We also assign an access window to both DOM0 and DOMU. The access window provides a dedicated GPU interface to each.

B.2.2 Device Tree

The virtualization stack provides additional modules which require a number of additional entries in the device tree. This section provides a detailed example for each module based on an Arm® Juno platform.

B.2.2.1 The host

With the following device tree entries, the host or DOM0 loads all of the mali_gpu_*.ko modules and applies the `sysfs` settings.

You can access the GPU through Access Window 0 after the probed `/dev/mali0` is available.



Make sure that this process works correctly before you create a guest VM.

System and Assign modules

The System and Assign modules must be configured in a privileged domain. In a production system, this may be done elsewhere in the system, for example by a *System Control Processor* (SCP), but for simplicity we do the configuration in DOM0.

The following device nodes handle system level operations and errors and configure GPU partitioning.

```
gpu_system@6e800000 {
    compatible = "arm,mali-gpu-system";
    reg = <0x0 0x6e800000 0x0 0x10000>;
    interrupts = <0 320 1>, <0 321 1>;
    /* PTM_UNCORRECTED_ERROR_IRQ and PTM_DEFERRED_ERROR_IRQ
     * controlled by PTM_SYSTEM for reporting all errors */
    interrupt-names = "PTM_UNCORRECTED_ERROR_IRQ",
                     "PTM_DEFERRED_ERROR_IRQ";
};

gpu_assign@6e810000 {
    compatible = "arm,mali-gpu-assign";
    reg = <0x0 0x6e810000 0x0 0x10000>;
};
```

Arbitration

You must place the Arbiter and its associated modules in a suitable domain or VM because they can affect the performance of all VMs using the GPU. For this example we use DOM0, where we have two partitions linked to the resource group through the `partition_control` and `partition_config` handles.

```
gpu_resource_group@6e0a0000 {
    compatible = "arm,mali-gpu-resource-group";
    reg = <0x0 0x6e0a0000 0x0 0x10000>;
    partition_control = <&gpu_partition_control_a 0>, <&gpu_partition_control_a 1>;
    partition_config = <&gpu_partition_config_a 0>, <&gpu_partition_config_a 1>;
    interrupts = <0 326 1>;
    /* PTM_GROUP_IRQ is a message signalling interrupt
     * one per group controlled by PTM_GROUP register page */
    interrupt-names = "PTM_GROUP_IRQ";
};

/* individual config/control for bus AXI-A */
gpu_partition_config_a_0:gpu_partition_config@6e020000 {
    compatible = "arm,mali-gpu-partition-config";
    reg = <0x0 0x6e020000 0x0 0x10000>;
};

gpu_partition_control_a_0:gpu_partition_control@6e030000 {
    compatible = "arm,mali-gpu-partition-control";
    reg = <0x0 0x6e030000 0x0 0x10000>;
    interrupts = <0 322 1>;
    /* PTM_PARTITION_IRQ is partition error signalling
     * interrupt one per partition controlled by
     * PARTITION_CONTROL register page */
};
```

```

        interrupt-names = "PTM_PARTITION_IRQ";
    };

    gpu_partition_config_a_1:gpu_partition_config@6e040000 {
        compatible = "arm,mali-gpu-partition-config";
        reg = <0x0 0x6e040000 0x0 0x10000>;
    };

    gpu_partition_control_a_1:gpu_partition_control@6e050000 {
        compatible = "arm,mali-gpu-partition-control";
        reg = <0x0 0x6e050000 0x0 0x10000>;
        interrupts = <0 323 1>;
        interrupt-names = "PTM_PARTITION_IRQ";
    };
};

```

Access Windows

We must define all of the access windows that are used by both DOM0 and DOMUs in the host device tree.

Any access windows that are to be passed through to guest VMs must have `xen,passthrough = <1>;` set in the DT node.

In this example, we keep AW0 in DOM0 and pass AW15 through to a guest VM.



We must specify the AxMMUSID StreamID for each of the access windows.

This allows the hypervisor to configure the *System Memory Management Unit* (SMMU) so that transactions from that access window have the correct guest *Intermediate Physical Address* (IPA) to guest *Physical Address* (PA) mapping applied.

```

gpu@6e0e0000 {
    compatible = "arm,mali-midgard";
    reg = <0x0 0x6e0e0000 0x0 0x1ffc0>;
    interrupts = <0 330 1>, <0 330 1>, <0 330 1>;
    interrupt-names = "JOB", "MMU", "GPU";
    arbiter_if = <&gpu_aw_message_a_0>;
    iommu = <&smmu_v3_0x00 &smmu_v3_0x01>;
};

gpu_aw_message_a_0:gpu_aw_message@6e0fffc0 {
    compatible = "arm,mali-gpu-aw-message";
    reg = <0x0 0x6e0fffc0 0x0 0x40>;
    interrupts = <0 330 1>;
    interrupt-names = "PTM_MESSAGE";
};

gpu@6e2c0000 {
    xen,passthrough = <1>;
    status = "disabled";
    compatible = "arm,mali-midgard";
    reg = <0x0 0x6e2c0000 0x0 0x1ffc0>;
    interrupts = <0 345 1>, <0 345 1>, <0 345 1>;
    interrupt-names = "JOB", "MMU", "GPU";
    arbiter_if = <&gpu_aw_message_a_15>;
    iommu = <&smmu_v3_0x1E &smmu_v3_0x1F>;
};

gpu_aw_message_a_15:gpu_aw_message@6e2dffc0 {
    xen,passthrough = <1>;
    status = "disabled";
};

```

```
compatible = "arm,mali-gpu-aw-message";
reg = <0x0 0x6e2dfc0 0x0 0x40>;
interrupts = <0 345 1>;
interrupt-names = "PTM_MESSAGE";
};
```

Memory Management Units

The *Memory Management Unit* (MMU) In Mali GPUs has a single stage of address mapping, which is used for stage 1 guest virtual address to guest PA mapping.

The hypervisor must configure an external SMMU to perform the stage 2 guest PA to machine PA mapping.

In this example, we use an MMU-600 as the SMMU.

```
smmu_v3:smmu@6F500000 {
    compatible = "arm,smmu-v3";
    reg = <0x0 0x6f500000 0x0 0x20000>;
    #global-interrupts = <0x1>;
    msi-parent = <&v2m 0>;
    interrupts = <0x0 0xa8 0x3>;
    interrupt-names = "combined";
    #iommu-cells = <0x1>;
};
```

B.2.2.2 The guest

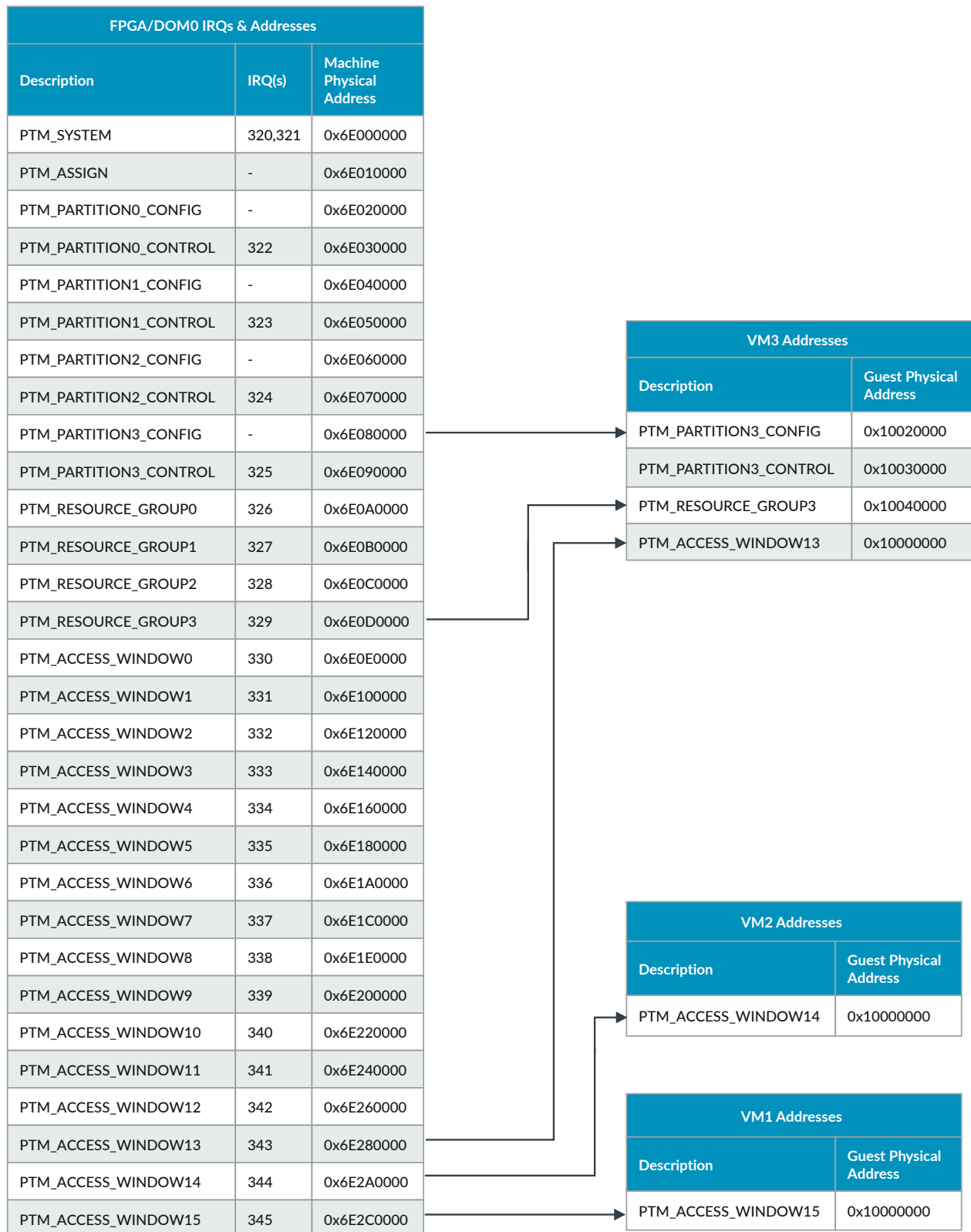
Now that we have a host device tree with appropriate device tree nodes, we can pass the device to a guest or DOMU VM.

The key concept is that we are mapping from a real *Physical Address* (PA) to a guest *Intermediate Physical Address* (IPA). We now have 3 addresses: PA, IPA, and virtual address. The guest kernel handled virtual address to IPA mapping as stage 1 and the hypervisor manages IPA to PA mapping as stage 2.

Starting from a working Xen setup, you already have a `guest.cfg`. We are modifying the configuration to map the GPU IRQ and I/O registers into a guest.

High level mapping

The following diagram shows an example address map for a DOM0 and 3 example DOMU VMs. VM3 is running a second arbiter instance.

Figure B-3: Example address map of the host and 3 guest virtual machines

- An access window is 2x 64K pages and includes both the GPU registers and AW_MESSAGE registers.
- The reason for using 0x1000 0000 as the base address rather than 0x6e00 0000 is that the latter is already in use by the guest for another non-GPU peripheral.
- The guest mappings reuse the same virtual guest PA space. This means that the guest device trees can be the same, but with modified IRQs.
- The example is only for AXI-A. Resources may be moved using mali_gpu_assign.ko, so take this into consideration.

Creating guest.cfg entries

The following tables describe the mappings in more detail, and explain how we go from the mapping above to entries in the `guest.cfg` file.

Table B-1: PTM_ACCESS_WINDOW13

PTM_ACCESS_WINDOW13, PTM_RESOURCE_GROUP3, PTM_PARTITION3_CONFIG/CONTROL (VM 3 guest.cfg)				
	DOM0	Guest	Entry in guest.cfg	Notes
IOMEM	0x6E280000 - 0x6E29FFFF 32 x 4k pages	0x10000000 - 0x1001FFFF 32 x 4k pages	iomem="0x6e280,0x20@0x10000"	Access Window 13 (GPU and AW regs) in a single mapping.
IOMEM	0x6E080000 - 0x6E09FFFF 32 x 4k pages	0x10020000 - 0x1003FFFF 32 x 4k pages	iomem="0x6e080,0x20@0x10020"	Partition Config and Control Combined.
IOMEM	0x6E0D0000 - 0x6E0DFFFF 16 x 4k pages	0x10040000 - 0x1004FFFF 16 x 4k pages	iomem="0x6e0d0,0x10@0x10040"	Resource Group guest_partial.dtb must use phandles to link with config/control.
IRQ(s)	329, 325, 343	329, 325, 343	irqs = [361, 357, 375]	Can not route an IRQ to multiple domains. guest.cfg is always offset from DT by +32.
DT Node(s)	/path/to/gpu@2a0000/ path/to/ gpu_aw_message@2bffc0	/gpu@102c0000 /gpu_aw_message@102dffc0	dtdev = [/path/to/gpu@2a0000, /path/to/ gpu_aw_message@2bffc0]	No need to move nodes to the top level, just need <code>xen, passthrough=1</code> and the correct path.

Table B-2: PTM_ACCESS_WINDOW14

PTM_ACCESS_WINDOW14 (VM 2 guest.cfg)				
	DOM0	Guest	Entry in guest.cfg	Notes
IOMEM	0x6E2A0000 - 0x6E2BFFFF 32 x 4k pages	0x10000000 - 0x1001FFFF 32 x 4k pages	iomem="0x6e2a0,0x20@0x10000"	Access Window 14 (GPU and AW regs) in a single mapping.
IRQ(s)	344	344	irqs = [376]	Can not route an IRQ to multiple domains. guest.cfg is always offset from DT by +32.

PTM_ACCESS_WINDOW14 (VM 2 guest.cfg)				
	DOM0	Guest	Entry in guest.cfg	Notes
DT Node(s)	/path/to/gpu@2a0000/ path/to/ gpu_aw_message@2bffc0	/gpu@102c0000/ gpu_aw_message@102dffc0	dtdev = [/path/to/gpu@2a0000, /path/to/ gpu_aw_message@2bffc0]	No need to move nodes to the top level, just need <code>xen, passthrough=1</code> and the correct path.

Table B-3: PTM_ACCESS_WINDOW15

PTM_ACCESS_WINDOW15 (VM 1 guest.cfg)				
	DOM0	Guest	Entry in guest.cfg	Notes
IOMEM	0x6E2C0000 - 0x6E2DFFFF 32 x 4k pages	0x10000000 - 0x1001FFFF 32 x 4k pages	iomem="0x6e2c0,0x20@0x10000"	Access Window 15 (GPU and AW regs) in a single mapping.
IRQ(s)	345	345	irqs = [377]	Can not route an IRQ to multiple domains. guest.cfg is always offset from DT by +32.
DT Node(s)	/path/to/gpu@2c0000 /path/to/ gpu_aw_message@2dffc0	/gpu@102c0000 /gpu_aw_message@102dffc0	dtdev = [/path/to/gpu@2c0000, /path/to/ gpu_aw_message@2dffc0]	No need to move nodes to the top level, just need <code>xen, passthrough=1</code> and the correct path.

For more details on the `guest.cfg` and `.dts` entries required for Xen, see <https://xenbits.xen.org/docs/4.14-testing/misc/arm/passthrough.txt>.

For this example, we use the VM1 from the previous diagram, which results in the following `guest.cfg`:

```
name = "debian1"
extra = " earlyprintk=xenboot console=hvc0 rw ip=dhcp rootwait debug
systemd.log_target=null user_debug=31 loglevel=9"
# Guest root filesystem device (from guest point of view)
root = "/dev/xvda1"
# Disk that will be used by the guest (set by manager when guest is created): disk 1
disk = ['tap:aio:juno.img,xvda,w']
kernel = "Image"
device_tree = "juno-bifrost-ptm-partial-01.dtb"
vfb = [ 'type=vnc,vncdisplay=1' ]
vif = ['script=debian1-net']
dtdev = [ "/gpu@6e2c0000" ]
irqs = [ 377, 361, 357 ]
# Map Access Window 15 guest locations
iomem = [ "0x6e2c0,0x20@0x10000" ]
memory = 2048
vcpus = 1
```

Creating a partial device tree

Finally, we must create a partial device tree exposing the mapped device to the guest:

```
/dts-v1/;
/ {
    passthrough {
        compatible = "simple-bus";
        ranges;
```

```

        #address-cells = <0x2>;
        #size-cells = <0x2>;

        /*
         * Generic GPU and Access Window Entry
         * Update the IRQ to match the ones in guest.cfg.
         * (This will only work with AW15)
         */
        gpu@0x10000000 {
            compatible = "arm,mali-midgard";
            reg = <0x0 0x10000000 0x0 0x1ffc0>;
            interrupts = <0 345 1>, <0 345 1>, <0 345 1>;
            interrupt-names = "JOB", "MMU", "GPU";
            arbiter_if = <&gpu_aw_message>;
        };

        gpu_aw_message:gpu_aw_message@0x1001ffc0 {
            compatible = "arm,mali-gpu-aw-message";
            reg = <0x0 0x1001ffc0 0x0 0x40>;
            interrupts = <0 345 1>;
            interrupt-names = "PTM_MESSAGE";
        };
    };
};

```

**Note**

This example does not cover putting the arbiter and related modules into a guest VM. Although it is not covered in this example, it is possible to put the arbiter and related modules into a guest VM using the preceding process as above but with different IRQs and IO ranges.

B.2.3 Flows

In this section, there are examples of how to correctly insert the modules at boot, configure the arbiter, and create a *Virtual Machine* (VM).

Boot or configuration

At boot time, based on the device tree previously defined, the hypervisor starts the DOM0. After the host is booted, you must load the virtualization stack and configure sysfs:

```

insmod mali_gpu_power.ko
insmod mali_gpu_system.ko
# Resource Group 0 with 2 slices, 2 partitions and 2 Access Windows
insmod mali_gpu_assign.ko ptm_config=A:S0:S1:P0:P1:W0:W15
insmod mali_gpu_partition_config.ko
insmod mali_gpu_partition_control.ko
# Longer yield timeout when running on 50MHz FPGA
insmod mali_arbiter.ko yield_timeout=1000
insmod mali_gpu_resource_group.ko
insmod mali_gpu_aw.ko

# Configure the Arbiter to enable the slices (0 & 1) and access windows (0 & 15)
echo "0x3" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/active_slices
echo "0x8001" > /sys/bus/platform/devices/6e0a0000.gpu_resource_group/arbiter/
partitions/partition0/assigned_access_windows

insmod mali_kbase.ko

```


This configures the GPU hardware and arbiter ready for AW0 and AW15 to be time sliced on partition 0, which contains slices 0 and 1.

Virtual Machine creation

Creating a VM is done in the same way as the existing system using the configuration file generated previously.

```
xl create guest.cfg
```

This creates the guest VM. Xen applies the IPA to PA mappings in stage 2 of the CPUs MMU and in stage 2 of the sMMUv3 under the GPU. Once configured, all transactions are handled through AxMMUSID Stream ID and sMMUv3 hardware.

After the VM is booted, the access window and Kbase modules must be inserted in the guest VM.

```
insmod mali_gpu_aw.ko  
insmod mali_kbase.ko
```



There is no need to load other modules unless the use case requires an arbiter in the guest VM.

Virtual Machine switching

VM switching happens transparently to the hypervisor.

When yielding the GPU, the arbiter requests that a VM stops using the GPU. When the VM responds that it has finished using the GPU, the arbiter closes the access window and assigns the GPU to the next waiting access window.

As part of this handover, the GPU instance is reset and the new AxMMUSID StreamID is associated with any new transactions from the new access window. Reset and AxMMUSID is all handled by the hardware. There is no communication with the hypervisor required.

Appendix C Partition manager

GPUs that support hardware virtualization contain a partition manager. This chapter introduces partition manager registers.



The Mali™-G78AE supports hardware virtualization.

C.1 Register page PARTITION_MANAGER

Partition manager control registers.

Control over the Partition Manager and access to the GPU partitions.

Address	Name	Usage	Access
0x0000 - 0xFFFFC	PTM_SYSTEM	Partition manager system registers	rw
0x10000 - 0x1FFFFC	PTM_ASSIGN	Partition manager assignment registers	rw
0x20000 - 0x2FFFFC	PTM_PARTITION0_CONFIG	Partition 0 configuration	rw
0x30000 - 0x3FFFFC	PTM_PARTITION0_CONTROL	Partition 0 control	rw
0x40000 - 0x4FFFFC	PTM_PARTITION1_CONFIG	Partition 1 configuration	rw
0x50000 - 0x5FFFFC	PTM_PARTITION1_CONTROL	Partition 1 control	rw
0x60000 - 0x6FFFFC	PTM_PARTITION2_CONFIG	Partition 2 configuration	rw
0x70000 - 0x7FFFFC	PTM_PARTITION2_CONTROL	Partition 2 control	rw
0x80000 - 0x8FFFFC	PTM_PARTITION3_CONFIG	Partition 3 configuration	rw
0x90000 - 0x9FFFFC	PTM_PARTITION3_CONTROL	Partition 3 control	rw
0xA0000 - 0xAFFFFC	PTM_RESOURCE_GROUP0	Resource group 0	rw
0xB0000 - 0xBFFFFC	PTM_RESOURCE_GROUP1	Resource group 1	rw
0xC0000 - 0xCFFFFC	PTM_RESOURCE_GROUP2	Resource group 2	rw
0xD0000 - 0xDFFFFC	PTM_RESOURCE_GROUP3	Resource group 3	rw
0xE0000 - 0xFFFFC	PTM_ACCESS_WINDOW0	Access window 0 to GPU control registers	rw
0x100000 - 0x11FFFFC	PTM_ACCESS_WINDOW1	Access window 1 to GPU control registers	rw
0x120000 - 0x13FFFFC	PTM_ACCESS_WINDOW2	Access window 2 to GPU control registers	rw
0x140000 - 0x15FFFFC	PTM_ACCESS_WINDOW3	Access window 3 to GPU control registers	rw
0x160000 - 0x17FFFFC	PTM_ACCESS_WINDOW4	Access window 4 to GPU control registers	rw
0x180000 - 0x19FFFFC	PTM_ACCESS_WINDOW5	Access window 5 to GPU control registers	rw
0x1A0000 - 0x1BFFFFC	PTM_ACCESS_WINDOW6	Access window 6 to GPU control registers	rw
0x1C0000 - 0x1DFFFFC	PTM_ACCESS_WINDOW7	Access window 7 to GPU control registers	rw
0x1E0000 - 0x1FFFFC	PTM_ACCESS_WINDOW8	Access window 8 to GPU control registers	rw

Address	Name	Usage	Access
0x200000 - 0x21FFFC	PTM_ACCESS_WINDOW9	Access window 9 to GPU control registers	rw
0x220000 - 0x23FFFC	PTM_ACCESS_WINDOW10	Access window 10 to GPU control registers	rw
0x240000 - 0x25FFFC	PTM_ACCESS_WINDOW11	Access window 11 to GPU control registers	rw
0x260000 - 0x27FFFC	PTM_ACCESS_WINDOW12	Access window 12 to GPU control registers	rw
0x280000 - 0x29FFFC	PTM_ACCESS_WINDOW13	Access window 13 to GPU control registers	rw
0x2A0000 - 0x2BFFFC	PTM_ACCESS_WINDOW14	Access window 14 to GPU control registers	rw
0x2C0000 - 0x2DFFFC	PTM_ACCESS_WINDOW15	Access window 15 to GPU control registers	rw

List of registers

PTM_SYSTEM (0x0000 - 0xFFFC)

Partition manager system registers.

System controls. Details on the [PTM_SYSTEM](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ASSIGN (0x10000 - 0x1FFFC)

Partition manager assignment registers.

Assignment of resources. Details on the [PTM_ASSIGN](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION0_CONFIG (0x20000 - 0x2FFFC)

Partition 0 configuration.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONFIG](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION0_CONTROL (0x30000 - 0x3FFFC)

Partition 0 control.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION1_CONFIG (0x40000 - 0x4FFFC)

Partition 1 configuration.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONFIG](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION1_CONTROL (0x50000 - 0x5FFFC)

Partition 1 control.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION2_CONFIG (0x60000 - 0x6FFFC)

Partition 2 configuration.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONFIG](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION2_CONTROL (0x70000 - 0x7FFFC)

Partition 2 control.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION3_CONFIG (0x80000 - 0x8FFFC)

Partition 3 configuration.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONFIG](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_PARTITION3_CONTROL (0x90000 - 0x9FFFC)

Partition 3 control.

Configuration controls for each partition. Details on the [PTM_PARTITION_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_RESOURCE_GROUP0 (0xA0000 - 0xAFFFC)

Resource group 0.

Control registers for each group. Details on the [PTM_RESOURCE_GROUP](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_RESOURCE_GROUP1 (0xB0000 - 0xBFFFC)

Resource group 1.

Control registers for each group. Details on the [PTM_RESOURCE_GROUP](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_RESOURCE_GROUP2 (0xC0000 - 0xCFFFC)

Resource group 2.

Control registers for each group. Details on the [PTM_RESOURCE_GROUP](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_RESOURCE_GROUP3 (0xD0000 - 0xDFFFC)

Resource group 3.

Control registers for each group. Details on the [PTM_RESOURCE_GROUP](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW0 (0xE0000 - 0xFFFFC)

Access window 0 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW1 (0x100000 - 0x11FFFC)

Access window 1 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW2 (0x120000 - 0x13FFFC)

Access window 2 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW3 (0x140000 - 0x15FFFC)

Access window 3 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW4 (0x160000 - 0x17FFFC)

Access window 4 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW5 (0x180000 - 0x19FFFC)

Access window 5 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW6 (0x1A0000 - 0x1BFFFC)

Access window 6 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW7 (0x1C0000 - 0x1DFFFC)

Access window 7 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW8 (0x1E0000 - 0x1FFFC)

Access window 8 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW9 (0x200000 - 0x21FFFC)

Access window 9 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW10 (0x220000 - 0x23FFFC)

Access window 10 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW11 (0x240000 - 0x25FFFC)

Access window 11 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW12 (0x260000 - 0x27FFFC)

Access window 12 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW13 (0x280000 - 0x29FFFC)

Access window 13 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW14 (0x2A0000 - 0x2BFFFC)

Access window 14 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ACCESS_WINDOW15 (0x2C0000 - 0x2DFFFC)

Access window 15 to GPU control registers.

An address range giving access to GPU and message registers. Details on the [PTM_ACCESS_WINDOW](#) sub-page.

Access to this Register is restricted to mode [rw](#).

C.1.1 Register sub-page PTM_ACCESS_WINDOW

Layout

Access window to GPU and message registers.

Address	Name	Usage	Access
0x0000 - 0x1FFBC	PTM_AW_GPU	Access window GPU registers	rw
0x1FFC0	PTM_ID	Partition Manager and GPU ID	ro
0x1FFC4	PTM_AW_IRQ_RAWSTAT	Status of access window interrupts before masking	ro
0x1FFC8	PTM_AW_IRQ_CLEAR	Clear incoming message	rw
0x1FFCC	PTM_AW_IRQ_MASK	Enable and disable the message interrupt	rw
0x1FFD0	PTM_AW_IRQ_STATUS	Status of the message after masking	ro
0x1FFD4	PTM_AW_IRQ_INJECTION	Inject interrupt before masking	rw
0x1FFD8 - 0x1FFF4	PTM_AW_MESSAGE	Access window message registers	rw
0x1FFF8 - 0x1FFFC	Reserved	-	-

List of registers

PTM_AW_GPU (0x0000 - 0x1FFBC)

Access window GPU registers.

An address range giving access to the GPU registers. Details on the [PTM_AW_GPU](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_ID (0x1FFC0)

Partition Manager and GPU ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_ACCESS_WINDOW.PTM_ID](#) layout

Field VERSION_STATUS

Contains the status of the GPU release.

[PTM_ACCESS_WINDOW.PTM_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field VERSION_MINOR

Contains the minor release number of the GPU (the P part of an RnP_n release number).

[PTM_ACCESS_WINDOW.PTM_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field VERSION_MAJOR

Contains the major release number of the GPU (the R part of an RnP_n release number).

[PTM_ACCESS_WINDOW.PTM_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field PRODUCT_MAJOR

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_ACCESS_WINDOW.PTM_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_REV

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_ACCESS_WINDOW.PTM_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_MINOR

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_ACCESS_WINDOW.PTM_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field ARCH_MAJOR

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_ACCESS_WINDOW.PTM_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_AW_IRQ_RAWSTAT (0x1FFC4)

Status of access window interrupts before masking by PTM_AW_IRQ_MASK.

Access window

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	MESSAGE	Set on receipt of a message	0
1	INVALID_ACCESS	Invalid access (attempt to access disabled window)	0
2	GPU_IRQ	Interrupt from GPU (reads as zero while window is disabled)	0
3	GPU_JOB	Job interrupt from GPU (reads as zero while window is disabled)	0
4	GPU_MMU	MMU interrupt from GPU (reads as zero while window is disabled)	0
5	GPU_EVENT	Event interrupt from GPU (reads as zero while window is disabled)	0
31:6	Reserved (zero)	-	-

Register [PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT layout

Field MESSAGE

Set on receipt of a message.

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT.MESSAGE (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE values

Field INVALID_ACCESS

Invalid access (attempt to access disabled window).

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT.INVALID_ACCESS (I) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_ACCESS values

Field GPU_IRQ

Interrupt from GPU (reads as zero while window is disabled).

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT.GPU_IRQ (GI) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No interrupt
1	Interrupt

Field GPU_IRQ values

Field GPU_JOB

Job interrupt from GPU (reads as zero while window is disabled).

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT.GPU_JOB (GJ) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No interrupt
1	Interrupt

Field GPU_JOB values

Field GPU_MMU

MMU interrupt from GPU (reads as zero while window is disabled).

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_RAWSTAT.GPU_MMU (GM) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No Interrupt
1	Interrupt

Field GPU_MMU values

Field GPU_EVENT

Event interrupt from GPU (reads as zero while window is disabled).

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_RAWSTAT](#).GPU_EVENT (GE) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No Interrupt
1	Interrupt

Field GPU_EVENT values

PTM_AW_IRQ_CLEAR (0x1FFC8)

Clear incoming message.

Write '1' to clear the status; this register always reads as zero. GPU Interrupts should be cleared at source.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE	Clear message status	0
1	INVALID_ACCESS	Clear invalid access interrupt	0
31:2	Reserved (zero)	-	-

Register [PTM_ACCESS_WINDOW.PTM_AW_IRQ_CLEAR](#) layout

Field MESSAGE

Clear message status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_CLEAR](#).MESSAGE (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear flag

Field MESSAGE values

Field INVALID_ACCESS

Clear invalid access interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_CLEAR](#).INVALID_ACCESS (I) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Do nothing
1	Clear flag

Field INVALID_ACCESS values

PTM_AW_IRQ_MASK (0x1FFCC)

Enable and disable the message interrupt.

Set to '1' to enable the interrupt.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE	Enable message interrupt	0
1	INVALID_ACCESS	Enable invalid access interrupt	0
2	GPU_IRQ	Enable GPU Interrupt	0
3	GPU_JOB	Enable GPU job interrupt	0
4	GPU_MMU	Enable GPU MMU interrupt	0
5	GPU_EVENT	Enable GPU event interrupt	0
31:6	Reserved (zero)	-	-

Register [PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_MASK layout

Field MESSAGE

Enable message interrupt.

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_MASK.MESSAGE (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE values

Field INVALID_ACCESS

Enable invalid access interrupt.

[PTM_ACCESS_WINDOW](#).PTM_AW_IRQ_MASK.INVALID_ACCESS (I) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_ACCESS values

Field GPU_IRQ

Enable GPU Interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_MASK.GPU_IRQ](#) (GI) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_IRQ values

Field GPU_JOB

Enable GPU job interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_MASK.GPU_JOB](#) (GJ) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_JOB values

Field GPU_MMU

Enable GPU MMU interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_MASK.GPU_MMU](#) (GM) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_MMU values

Field GPU_EVENT

Enable GPU event interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_MASK.GPU_EVENT](#) (GE) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_EVENT values

PTM_AW_IRQ_STATUS (0x1FFD0)

Status of the access window interrupts after masking by PTM_AW_IRQ_MASK..

Report of interrupt status after masking: an interrupt is raised if the bit is '1'.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	MESSAGE	Message interrupt status	0
1	INVALID_ACCESS	Invalid access interrupt status	0
2	GPU_IRQ	GPU Interrupt status	0
3	GPU_JOB	Inactive	0
4	GPU_MMU	GPU MMU interrupt status	0
5	GPU_EVENT	GPU event interrupt status	0
31:6	Reserved (zero)	-	-

Register [PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS](#) layout

Field MESSAGE

Message interrupt status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS.MESSAGE](#) (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field MESSAGE values

Field INVALID_ACCESS

Invalid access interrupt status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS.INVALID_ACCESS](#) (I) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field INVALID_ACCESS values

Field GPU_IRQ

GPU Interrupt status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS.GPU_IRQ](#) (GI) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field GPU_IRQ values

Field GPU_JOB

Inactive.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS.GPU_JOB](#) (GJ) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field GPU_JOB values

Field GPU_MMU

GPU MMU interrupt status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS](#).GPU_MMU (GM) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field GPU_MMU values

Field GPU_EVENT

GPU event interrupt status.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_STATUS](#).GPU_EVENT (GE) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Inactive
1	Active

Field GPU_EVENT values

PTM_AW_IRQ_INJECTION (0x1FFD4)

Inject interrupt before masking.

Inject interrupts to test interrupt handlers are correctly configured.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE	Inject message interrupt	0
1	INVALID_ACCESS	Inject message interrupt	0
2	GPU_IRQ	Inject GPU Interrupt	0
3	GPU_JOB	Disable	0
4	GPU_MMU	Inject GPU MMU interrupt	0
5	GPU_EVENT	Inject GPU event interrupt	0
31:6	Reserved (zero)	-	-

Register [PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#) layout

Field MESSAGE

Inject message interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).MESSAGE (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE values

Field INVALID_ACCESS

Inject message interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).INVALID_ACCESS (I) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_ACCESS values

Field GPU_IRQ

Inject GPU Interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).GPU_IRQ (GI) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_IRQ values

Field GPU_JOB

Disable.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).GPU_JOB (GJ) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_JOB values

Field GPU_MMU

Inject GPU MMU interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).GPU_MMU (GM) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_MMU values

Field GPU_EVENT

Inject GPU event interrupt.

[PTM_ACCESS_WINDOW.PTM_AW_IRQ_INJECTION](#).GPU_EVENT (GE) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GPU_EVENT values

PTM_AW_MESSAGE (0x1FFD8 - 0x1FFF4)

Access window message registers. A set of registers used for communication between an instance of the driver and system software. Details on the PTM_MESSAGE sub-page.

Access to this Register is restricted to mode rw.

C.1.2 Register sub-page PTM_ASSIGN

Layout

Assignment of resources to groups and groups to buses.

Address	Name	Usage	Access
0x0000	PTM_DEVICE_ID	Partition Manager ID	ro
0x0004	PTM_UNIT_FEATURES	Partition Manager features	ro
0x0008	PTM_SLICE_FEATURES	Slice features	ro
0x000C - 0x0010	PTM_SLICE_CORES	Cores per slice	ro
0x0014 - 0x003C	Reserved		
0x0040	PTM_RESOURCE_GROUP_BUS	Assignment of buses to resource groups	rw
0x0044	PTM_PARTITION_RESOURCE_GROUP	Assignment of partitions into resource groups	rw
0x0048	PTM_SLICE_RESOURCE_GROUP	Assignment of slices into resource groups	rw
0x004C	PTM_SLICE_ISOLATION_STATUS	Status of isolation between slices	ro
0x0050	PTM_SLICE_ISOLATION_SET	Control isolation between slices	rw
0x0054 - 0x007C	Reserved		
0x0080 - 0x008C	PTM_AW_RESOURCE_GROUP	Assignment of access windows into resource groups	rw
0x0090 - 0xFFFFD	Reserved	-	-

List of registers

PTM_DEVICE_ID (0x0000)

Partition Manager ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_ASSIGN.PTM_DEVICE_ID](#) layout

Field **VERSION_STATUS**

Contains the status of the GPU release.

[PTM_ASSIGN.PTM_DEVICE_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field **VERSION_MINOR**

Contains the minor release number of the GPU (the P part of an RnPn release number).

[PTM_ASSIGN.PTM_DEVICE_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **VERSION_MAJOR**

Contains the major release number of the GPU (the R part of an RnPn release number).

[PTM_ASSIGN.PTM_DEVICE_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **PRODUCT_MAJOR**

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_ASSIGN.PTM_DEVICE_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field **ARCH_REV**

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_ASSIGN.PTM_DEVICE_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field **ARCH_MINOR**

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_ASSIGN.PTM_DEVICE_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field **ARCH_MAJOR**

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_ASSIGN.PTM_DEVICE_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_UNIT_FEATURES (0x0004)

Partition Manager features.

Identifies the partition manager features.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITIONS	Total number of partitions in the system	Implementation defined	1	4
7:4	Reserved (zero)	-	-	-	-
13:8	ACCESS_WINDOWS	Total number of access windows in the system	Implementation defined	1	16
31:14	Reserved (zero)	-	-	-	-

Register [PTM_ASSIGN.PTM_UNIT_FEATURES](#) layout

Field PARTITIONS

Total number of partitions in the system.

[PTM_ASSIGN.PTM_UNIT_FEATURES.PARTITIONS](#) (P) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 1 to 4 inclusive. Its default value is Implementation defined.

Field ACCESS_WINDOWS

Total number of access windows in the system.

[PTM_ASSIGN.PTM_UNIT_FEATURES.ACCESS_WINDOWS](#) (W) is stored in bits [13:8] and is a 6-bit unsigned integer. Its value must lie in the range from 1 to 16 inclusive. Its default value is Implementation defined.

PTM_SLICE_FEATURES (0x0008)

Slice features.

Reports features of each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
7:0	CORE_MASK_STRIDE	Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice)	4
8	SLICE0_TILER	Type of tiler in slice 0	Implementation defined
9	SLICE1_TILER	Type of tiler in slice 1	Implementation defined

Bits	Name	Usage	Default
10	SLICE2_TILER	Type of tiler in slice 2	Implementation defined
11	SLICE3_TILER	Type of tiler in slice 3	Implementation defined
12	SLICE4_TILER	Type of tiler in slice 4	Implementation defined
13	SLICE5_TILER	Type of tiler in slice 5	Implementation defined
14	SLICE6_TILER	Type of tiler in slice 6	Implementation defined
15	SLICE7_TILER	Type of tiler in slice 7	Implementation defined
31:16	Reserved (zero)	-	-

Register [PTM_ASSIGN.PTM_SLICE_FEATURES](#) layout

Field **CORE_MASK_STRIDE**

Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice).

[PTM_ASSIGN.PTM_SLICE_FEATURES.CORE_MASK_STRIDE](#) (S) is stored in bits [7:0] and is a 8-bit unsigned integer. Its default value is 4.

Field **SLICE0_TILER**

Type of tiler in slice 0.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE0_TILER](#) (T0) is stored in bit [8] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE0_TILER values

Field **SLICE1_TILER**

Type of tiler in slice 1.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE1_TILER](#) (T1) is stored in bit [9] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE1_TILER values

Field **SLICE2_TILER**

Type of tiler in slice 2.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE2_TILER](#) (T2) is stored in bit [10] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE2_TILER values

Field SLICE3_TILER

Type of tiler in slice 3.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE3_TILER](#) (T3) is stored in bit [11] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE3_TILER values

Field SLICE4_TILER

Type of tiler in slice 4.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE4_TILER](#) (T4) is stored in bit [12] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE4_TILER values

Field SLICE5_TILER

Type of tiler in slice 5.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE5_TILER](#) (T5) is stored in bit [13] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE5_TILER values

Field SLICE6_TILER

Type of tiler in slice 6.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE6_TILER](#) (T6) is stored in bit [14] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE6_TILER values

Field SLICE7_TILER

Type of tiler in slice 7.

[PTM_ASSIGN.PTM_SLICE_FEATURES.SLICE7_TILER](#) (T7) is stored in bit [15] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE7_TILER values

PTM_SLICE_CORES (0x000C - 0x0010)

Cores per slice.

Identifies the number of cores in each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
7:0	CORES0	Number of shader cores in slice 0	Implementation defined	0	3
15:8	CORES1	Number of shader cores in slice 1	Implementation defined	0	3
23:16	CORES2	Number of shader cores in slice 2	Implementation defined	0	3
31:24	CORES3	Number of shader cores in slice 3	Implementation defined	0	3
39:32	CORES4	Number of shader cores in slice 4	Implementation defined	0	3
47:40	CORES5	Number of shader cores in slice 5	Implementation defined	0	3
55:48	CORES6	Number of shader cores in slice 6	Implementation defined	0	3
63:56	CORES7	Number of shader cores in slice 7	Implementation defined	0	3

Register [PTM_ASSIGN.PTM_SLICE_CORES](#) layout

Field CORES0

Number of shader cores in slice 0.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES0](#) (C0) is stored in bits [7:0] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES1

Number of shader cores in slice 1.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES1](#) (C1) is stored in bits [15:8] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES2

Number of shader cores in slice 2.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES2](#) (C2) is stored in bits [23:16] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES3

Number of shader cores in slice 3.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES3](#) (C3) is stored in bits [31:24] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES4

Number of shader cores in slice 4.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES4](#) (C4) is stored in bits [39:32] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES5

Number of shader cores in slice 5.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES5](#) (C5) is stored in bits [47:40] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES6

Number of shader cores in slice 6.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES6](#) (C6) is stored in bits [55:48] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES7

Number of shader cores in slice 7.

[PTM_ASSIGN.PTM_SLICE_CORES.CORES7](#) (C7) is stored in bits [63:56] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

PTM_RESOURCE_GROUP_BUS (0x0040)

Assignment of buses to resource groups.

Assigns each resource group to a bus; each group is accessed from one and only one bus.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
3:0	RESOURCE_GROUP0	Bus to which resource group 0 is assigned	BUS_A
7:4	RESOURCE_GROUP1	Bus to which resource group 1 is assigned	BUS_A
11:8	RESOURCE_GROUP2	Bus to which resource group 2 is assigned	BUS_A
15:12	RESOURCE_GROUP3	Bus to which resource group 3 is assigned	BUS_A
31:16	Reserved (zero)	-	-

Register [PTM_ASSIGN.PTM_RESOURCE_GROUP_BUS](#) layout

Field RESOURCE_GROUP0

Bus to which resource group 0 is assigned.

[PTM_ASSIGN.PTM_RESOURCE_GROUP_BUS.RESOURCE_GROUP0](#) (G0) is stored in bits [3:0] and is a 4-bit enumeration of type PTM_BUS_AB_ENUM. Its default value is BUS_A.

The field can contain the following values:

Value	Name	Meaning
0 (default)	BUS_A	Assign to Bus-A
1	BUS_B	Assign to Bus-B

Field RESOURCE_GROUP0 values

Field RESOURCE_GROUP1

Bus to which resource group 1 is assigned.

[PTM_ASSIGN.PTM_RESOURCE_GROUP_BUS.RESOURCE_GROUP1](#) (G1) is stored in bits [7:4] and is a 4-bit enumeration of type PTM_BUS_AB_ENUM. Its default value is BUS_A.

The field can contain the following values:

Value	Name	Meaning
0 (default)	BUS_A	Assign to Bus-A
1	BUS_B	Assign to Bus-B

Field RESOURCE_GROUP1 values

Field RESOURCE_GROUP2

Bus to which resource group 2 is assigned.

[PTM_ASSIGN.PTM_RESOURCE_GROUP_BUS.RESOURCE_GROUP2](#) (G2) is stored in bits [11:8] and is a 4-bit enumeration of type PTM_BUS_AB_ENUM. Its default value is BUS_A.

The field can contain the following values:

Value	Name	Meaning
0 (default)	BUS_A	Assign to Bus-A
1	BUS_B	Assign to Bus-B

Field RESOURCE_GROUP2 values

Field RESOURCE_GROUP3

Bus to which resource group 3 is assigned.

[PTM_ASSIGN.PTM_RESOURCE_GROUP_BUS.RESOURCE_GROUP3](#) (G3) is stored in bits [15:12] and is a 4-bit enumeration of type PTM_BUS_AB_ENUM. Its default value is BUS_A.

The field can contain the following values:

Value	Name	Meaning
0 (default)	BUS_A	Assign to Bus-A
1	BUS_B	Assign to Bus-B

Field RESOURCE_GROUP3 values

PTM_PARTITION_RESOURCE_GROUP (0x0044)

Assignment of partitions into resource groups.

Assigns each partition to a resource group; each partition is assigned to one and only group.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITION0	Group to which partition 0 is assigned	0	0	3
7:4	PARTITION1	Group to which partition 1 is assigned	0	0	3
11:8	PARTITION2	Group to which partition 2 is assigned	0	0	3
15:12	PARTITION3	Group to which partition 3 is assigned	0	0	3
31:16	Reserved (zero)	-	-	-	-

Register [PTM_ASSIGN.PTM_PARTITION_RESOURCE_GROUP](#) layout

Field **PARTITION0**

Group to which partition 0 is assigned.

[PTM_ASSIGN.PTM_PARTITION_RESOURCE_GROUP.PARTITION0](#) (P0) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **PARTITION1**

Group to which partition 1 is assigned.

[PTM_ASSIGN.PTM_PARTITION_RESOURCE_GROUP.PARTITION1](#) (P1) is stored in bits [7:4] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **PARTITION2**

Group to which partition 2 is assigned.

[PTM_ASSIGN.PTM_PARTITION_RESOURCE_GROUP.PARTITION2](#) (P2) is stored in bits [11:8] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **PARTITION3**

Group to which partition 3 is assigned.

[PTM_ASSIGN.PTM_PARTITION_RESOURCE_GROUP.PARTITION3](#) (P3) is stored in bits [15:12] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

PTM_SLICE_RESOURCE_GROUP (0x0048)

Assignment of slices into resource groups.

Assigns each slice to a resource group; each slice is assigned to one and only one group.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default	Min	Max
3:0	SLICE0	Group to which slice 0 is assigned	0	0	3
7:4	SLICE1	Group to which slice 1 is assigned	0	0	3
11:8	SLICE2	Group to which slice 2 is assigned	0	0	3

Bits	Name	Usage	Default	Min	Max
15:12	SLICE3	Group to which slice 3 is assigned	0	0	3
19:16	SLICE4	Group to which slice 4 is assigned	0	0	3
23:20	SLICE5	Group to which slice 5 is assigned	0	0	3
27:24	SLICE6	Group to which slice 6 is assigned	0	0	3
31:28	SLICE7	Group to which slice 7 is assigned	0	0	3

Register [PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP](#) layout

Field **SLICE0**

Group to which slice 0 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE0](#) (S0) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE1**

Group to which slice 1 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE1](#) (S1) is stored in bits [7:4] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE2**

Group to which slice 2 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE2](#) (S2) is stored in bits [11:8] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE3**

Group to which slice 3 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE3](#) (S3) is stored in bits [15:12] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE4**

Group to which slice 4 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE4](#) (S4) is stored in bits [19:16] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE5**

Group to which slice 5 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE5](#) (S5) is stored in bits [23:20] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field **SLICE6**

Group to which slice 6 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP.SLICE6](#) (S6) is stored in bits [27:24] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field SLICE7

Group to which slice 7 is assigned.

[PTM_ASSIGN.PTM_SLICE_RESOURCE_GROUP](#).SLICE7 (S7) is stored in bits [31:28] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

PTM_SLICE_ISOLATION_STATUS (0x004C)

Status of isolation between slices.

Reports isolation between adjacent slices.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	SLICE0	SLICE 0 isolation	0
1	SLICE1	SLICE 1 isolation	0
2	SLICE2	SLICE 2 isolation	0
3	SLICE3	SLICE 3 isolation	0
4	SLICE4	SLICE 4 isolation	0
5	SLICE5	SLICE 5 isolation	0
6	SLICE6	SLICE 6 isolation	0
7	SLICE7	SLICE 7 isolation	0
31:8	Reserved (zero)	-	-

Register [PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#) layout

Field SLICE0

SLICE 0 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE0 (S0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE0 values

Field SLICE1

SLICE 1 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE1 (S1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE1 values

Field SLICE2

SLICE 2 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE2 (S2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE2 values

Field SLICE3

SLICE 3 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE3 (S3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE3 values

Field SLICE4

SLICE 4 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE4 (S4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE4 values

Field SLICE5

SLICE 5 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE5 (S5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE5 values

Field SLICE6

SLICE 6 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE6 (S6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE6 values

Field SLICE7

SLICE 7 isolation.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_STATUS](#).SLICE7 (S7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not isolated from preceding slice
1	Isolated from preceding slice

Field SLICE7 values

PTM_SLICE_ISOLATION_SET (0x0050)

Control isolation between slices.

Enforces isolation between adjacent slices normally set to enforce isolation between partitions controlled by different arbiters.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	SLICE0	Isolation control of SLICE 0	0
1	SLICE1	Isolation control of SLICE 1	0
2	SLICE2	Isolation control of SLICE 2	0
3	SLICE3	Isolation control of SLICE 3	0
4	SLICE4	Isolation control of SLICE 4	0
5	SLICE5	Isolation control of SLICE 5	0
6	SLICE6	Isolation control of SLICE 6	0
7	SLICE7	Isolation control of SLICE 7	0
31:8	Reserved (zero)	-	-

Register [PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#) layout

Field SLICE0

Isolation control of SLICE 0.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE0 (S0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE0 values

Field SLICE1

Isolation control of SLICE 1.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE1 (S1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE1 values

Field SLICE2

Isolation control of SLICE 2.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE2 (S2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE2 values

Field SLICE3

Isolation control of SLICE 3.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE3 (S3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE3 values

Field SLICE4

Isolation control of SLICE 4.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE4 (S4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE4 values

Field SLICE5

Isolation control of SLICE 5.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE5 (S5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE5 values

Field SLICE6

Isolation control of SLICE 6.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE6 (S6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE6 values

Field SLICE7

Isolation control of SLICE 7.

[PTM_ASSIGN.PTM_SLICE_ISOLATION_SET](#).SLICE7 (S7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do not isolate from preceding slice
1	Isolate from preceding slice

Field SLICE7 values

PTM_AW_RESOURCE_GROUP (0x0080 - 0x008C)

Assignment of access windows into resource groups.

Assigns each window to a group; assigning does not enable the window.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default	Min	Max
3:0	AW0	Access window 0 group	0	0	3
7:4	AW1	Access window 1 group	0	0	3
11:8	AW2	Access window 2 group	0	0	3
15:12	AW3	Access window 3 group	0	0	3
19:16	AW4	Access window 4 group	0	0	3
23:20	AW5	Access window 5 group	0	0	3
27:24	AW6	Access window 6 group	0	0	3
31:28	AW7	Access window 7 group	0	0	3
35:32	AW8	Access window 8 group	0	0	3
39:36	AW9	Access window 9 group	0	0	3
43:40	AW10	Access window 10 group	0	0	3
47:44	AW11	Access window 11 group	0	0	3

Bits	Name	Usage	Default	Min	Max
51:48	AW12	Access window 12 group	0	0	3
55:52	AW13	Access window 13 group	0	0	3
59:56	AW14	Access window 14 group	0	0	3
63:60	AW15	Access window 15 group	0	0	3
127:64	Reserved (zero)	-	-	-	-

Register [PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#) layout

Field AW0

Access window 0 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW0](#) (AW0) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW1

Access window 1 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW1](#) (AW1) is stored in bits [7:4] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW2

Access window 2 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW2](#) (AW2) is stored in bits [11:8] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW3

Access window 3 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW3](#) (AW3) is stored in bits [15:12] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW4

Access window 4 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW4](#) (AW4) is stored in bits [19:16] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW5

Access window 5 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW5](#) (AW5) is stored in bits [23:20] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW6

Access window 6 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW6](#) (AW6) is stored in bits [27:24] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW7

Access window 7 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW7 (AW7) is stored in bits [31:28] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW8

Access window 8 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW8 (AW8) is stored in bits [35:32] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW9

Access window 9 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW9 (AW9) is stored in bits [39:36] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW10

Access window 10 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW10 (AW10) is stored in bits [43:40] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW11

Access window 11 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW11 (AW11) is stored in bits [47:44] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW12

Access window 12 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW12 (AW12) is stored in bits [51:48] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW13

Access window 13 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW13 (AW13) is stored in bits [55:52] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW14

Access window 14 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP](#).AW14 (AW14) is stored in bits [59:56] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

Field AW15

Access window 15 group.

[PTM_ASSIGN.PTM_AW_RESOURCE_GROUP.AW15](#) (AW15) is stored in bits [63:60] and is a 4-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is 0.

C.1.3 Register sub-page PTM_AW_GPU

Access window to partition GPU registers.

An address range giving access to the partition GPU registers. Enabled by PTM_AW_SET register; accessing a disabled window raises an error.

Layout

Address	Name	Usage
0x0000 - 0x1FFBC	Reserved	-

List of registers

C.1.4 Register sub-page PTM_BIST_CONTROL

Registers for controlling BIST.

BIST control registers for a slice; access is disabled unless the slice is enabled as a Requester or Completer and BIST is enabled for its partition.

Layout

Address	Name	Usage
0x0000 - 0x01FC	Reserved	-

List of registers

C.1.5 Register sub-page PTM_MESSAGE

Layout

Message registers.

Address	Name	Usage	Access
0x0000	PTM_INCOMING_MESSAGE0	Received message	ro
0x0004	PTM_INCOMING_MESSAGE1	Received message	ro
0x0008	PTM_OUTGOING_MESSAGE_STATUS	Message status	ro
0x000C	PTM_OUTGOING_MESSAGE0	Outgoing message	rw

0x0010	PTM_OUTGOING_MESSAGE1	Outgoing message	rw
0x0014 - 0x001C	Reserved	-	-

List of registers

PTM_INCOMING_MESSAGE0 (0x0000)

Received message.

Received message payload; a flag is set on receipt in PTM_AW_IRQ_RAWSTAT and an interrupt optionally raised. This register reads as zero when the flag is cleared..

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
31:0	MESSAGE	Message payload

Register [PTM_MESSAGE](#).PTM_INCOMING_MESSAGE0 layout

Field MESSAGE

Message payload.

[PTM_MESSAGE](#).PTM_INCOMING_MESSAGE0.MESSAGE (M) is stored in bits [31:0] and is a 32-bit unsigned integer.

PTM_INCOMING_MESSAGE1 (0x0004)

Received message.

Received message payload; a flag is set in PTM_AW_IRQ_RAWSTAT on receipt and an interrupt optionally raised. This register reads as zero when the flag is cleared.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
31:0	MESSAGE	Message payload

Register [PTM_MESSAGE](#).PTM_INCOMING_MESSAGE1 layout

Field MESSAGE

Message payload.

[PTM_MESSAGE](#).PTM_INCOMING_MESSAGE1.MESSAGE (M) is stored in bits [31:0] and is a 32-bit unsigned integer.

PTM_OUTGOING_MESSAGE_STATUS (0x0008)

Message status.

Status of a message sent via PTM_OUTGOING_MESSAGE register (reflects state of corresponding PTM_IRQ_RAWSTAT.MESSAGE).

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	MESSAGE_STATUS	Outgoing message status	0
31:1	Reserved (zero)	-	-

Register [PTM_MESSAGE](#).PTM_OUTGOING_MESSAGE_STATUS layout

Field MESSAGE_STATUS

Outgoing message status.

[PTM_MESSAGE](#).PTM_OUTGOING_MESSAGE_STATUS.MESSAGE_STATUS (S) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message pending
1	Message pending

Field MESSAGE_STATUS values

PTM_OUTGOING_MESSAGE0 (0x000C)

Outgoing message.

Outgoing message payload that becomes visible in receiving register following write to PTM_OUTGOING_MESSAGE1.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage
31:0	MESSAGE	Message payload

Register [PTM_MESSAGE](#).PTM_OUTGOING_MESSAGE0 layout

Field MESSAGE

Message payload.

[PTM_MESSAGE](#).PTM_OUTGOING_MESSAGE0.MESSAGE (M) is stored in bits [31:0] and is a 32-bit unsigned integer.

PTM_OUTGOING_MESSAGE1 (0x0010)

Outgoing message.

Outgoing message payload; write causes this register and PTM_OUTGOING_MESSAGE0 to become visible in receiving registers and sets outgoing status to pending.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage
31:0	MESSAGE	Message payload

Register [PTM_MESSAGE](#).PTM_OUTGOING_MESSAGE1 layout

Field MESSAGE

Message payload.

[PTM_MESSAGE.PTM_OUTGOING_MESSAGE1](#).MESSAGE (M) is stored in bits [31:0] and is a 32-bit unsigned integer.

C.1.6 Register sub-page PTM_PARTITION_CONFIG**Layout**

Registers for setting the configuration of a partition.

Address	Name	Usage	Access
0x0000	PTM_ID	Partition Manager ID	ro
0x0004	PTM_UNIT_FEATURES	Partition Manger features	ro
0x0008	PTM_SLICE_FEATURES	Features of each slice in the system	ro
0x000C - 0x0010	PTM_SLICE_CORES	Cores per slice	ro
0x0014	PTM_SLICE_MODE	Slice mode	ro
0x0018	PTM_SLICE_MODE_NEW	Set slice mode	rw
0x001C	PTM_SLICE_MODE_UPDATE	Update slice mode	rw
0x0020	PTM_SLICE_MODE_ACK	Acknowledge update of slice mode	rw
0x0024 - 0xFFFFD	Reserved	-	-

List of registers

PTM_ID (0x0000)

Partition Manager ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)

Bits	Name	Usage	Default
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_PARTITION_CONFIG.PTM_ID](#) layout

Field **VERSION_STATUS**

Contains the status of the GPU release.

[PTM_PARTITION_CONFIG.PTM_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field **VERSION_MINOR**

Contains the minor release number of the GPU (the P part of an RnPN release number).

[PTM_PARTITION_CONFIG.PTM_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **VERSION_MAJOR**

Contains the major release number of the GPU (the R part of an RnPn release number).

[PTM_PARTITION_CONFIG.PTM_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **PRODUCT_MAJOR**

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_PARTITION_CONFIG.PTM_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_REV

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONFIG.PTM_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_MINOR

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONFIG.PTM_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field ARCH_MAJOR

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONFIG.PTM_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_UNIT_FEATURES (0x0004)

Partition Manager features.

Identifies the partition manager features.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITIONS	Total number of partitions in the system	Implementation defined	1	4
7:4	Reserved (zero)	-	-	-	-
13:8	ACCESS_WINDOWS	Total number of access windows in the system	Implementation defined	1	16
31:14	Reserved (zero)	-	-	-	-

Register [PTM_PARTITION_CONFIG.PTM_UNIT_FEATURES](#) layout

Field **PARTITIONS**

Total number of partitions in the system.

[PTM_PARTITION_CONFIG.PTM_UNIT_FEATURES](#).PARTITIONS (P) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 1 to 4 inclusive. Its default value is Implementation defined.

Field **ACCESS_WINDOWS**

Total number of access windows in the system.

[PTM_PARTITION_CONFIG.PTM_UNIT_FEATURES](#).ACCESS_WINDOWS (W) is stored in bits [13:8] and is a 6-bit unsigned integer. Its value must lie in the range from 1 to 16 inclusive. Its default value is Implementation defined.

PTM_SLICE_FEATURES (0x0008)

Features of each slice in the system.

Reports features of each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
7:0	CORE_MASK_STRIDE	Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice)	4
8	SLICE0_TILER	Type of tiler in slice 0	Implementation defined
9	SLICE1_TILER	Type of tiler in slice 1	Implementation defined
10	SLICE2_TILER	Type of tiler in slice 2	Implementation defined
11	SLICE3_TILER	Type of tiler in slice 3	Implementation defined
12	SLICE4_TILER	Type of tiler in slice 4	Implementation defined
13	SLICE5_TILER	Type of tiler in slice 5	Implementation defined
14	SLICE6_TILER	Type of tiler in slice 6	Implementation defined
15	SLICE7_TILER	Type of tiler in slice 7	Implementation defined
31:16	Reserved (zero)	-	-

Register [PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES](#) layout

Field **CORE_MASK_STRIDE**

Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice).

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.CORE_MASK_STRIDE](#) (S) is stored in bits [7:0] and is a 8-bit unsigned integer. Its default value is 4.

Field **SLICE0_TILER**

Type of tiler in slice 0.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE0_TILER](#) (T0) is stored in bit [8] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE0_TILER values

Field **SLICE1_TILER**

Type of tiler in slice 1.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE1_TILER](#) (T1) is stored in bit [9] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE1_TILER values

Field **SLICE2_TILER**

Type of tiler in slice 2.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE2_TILER](#) (T2) is stored in bit [10] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE2_TILER values

Field **SLICE3_TILER**

Type of tiler in slice 3.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE3_TILER](#) (T3) is stored in bit [11] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE3_TILER values

Field SLICE4_TILER

Type of tiler in slice 4.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE4_TILER](#) (T4) is stored in bit [12] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE4_TILER values

Field SLICE5_TILER

Type of tiler in slice 5.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE5_TILER](#) (T5) is stored in bit [13] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE5_TILER values

Field SLICE6_TILER

Type of tiler in slice 6.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE6_TILER](#) (T6) is stored in bit [14] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE6_TILER values

Field SLICE7_TILER

Type of tiler in slice 7.

[PTM_PARTITION_CONFIG.PTM_SLICE_FEATURES.SLICE7_TILER](#) (T7) is stored in bit [15] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE7_TILER values

PTM_SLICE_CORES (0x000C - 0x0010)

Cores per slice.

Identifies the number of cores in each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
7:0	CORES0	Number of shader cores in slice 0	Implementation defined	0	3
15:8	CORES1	Number of shader cores in slice 1	Implementation defined	0	3
23:16	CORES2	Number of shader cores in slice 2	Implementation defined	0	3
31:24	CORES3	Number of shader cores in slice 3	Implementation defined	0	3
39:32	CORES4	Number of shader cores in slice 4	Implementation defined	0	3
47:40	CORES5	Number of shader cores in slice 5	Implementation defined	0	3
55:48	CORES6	Number of shader cores in slice 6	Implementation defined	0	3
63:56	CORES7	Number of shader cores in slice 7	Implementation defined	0	3

Register [PTM_PARTITION_CONFIG.PTM_SLICE_CORES](#) layout

Field **CORES0**

Number of shader cores in slice 0.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES0](#) (C0) is stored in bits [7:0] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field **CORES1**

Number of shader cores in slice 1.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES1](#) (C1) is stored in bits [15:8] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field **CORES2**

Number of shader cores in slice 2.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES2](#) (C2) is stored in bits [23:16] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field **CORES3**

Number of shader cores in slice 3.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES3](#) (C3) is stored in bits [31:24] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field **CORES4**

Number of shader cores in slice 4.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES4](#) (C4) is stored in bits [39:32] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field **CORES5**

Number of shader cores in slice 5.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES5](#) (C5) is stored in bits [47:40] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES6

Number of shader cores in slice 6.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES6](#) (C6) is stored in bits [55:48] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES7

Number of shader cores in slice 7.

[PTM_PARTITION_CONFIG.PTM_SLICE_CORES.CORES7](#) (C7) is stored in bits [63:56] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

PTM_SLICE_MODE (0x0014)

Slice mode.

Holds the slice configuration. Updated according to PTM_SLICE_MODE_NEW when writing PTM_SLICE_MODE_UPDATE. Fields for unassigned slices read zero.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
1:0	SLICE0_MODE	Current mode of slice 0	Disabled
3:2	SLICE1_MODE	Current mode of slice 1	Disabled
5:4	SLICE2_MODE	Current mode of slice 2	Disabled
7:6	SLICE3_MODE	Current mode of slice 3	Disabled
9:8	SLICE4_MODE	Current mode of slice 4	Disabled
11:10	SLICE5_MODE	Current mode of slice 5	Disabled
13:12	SLICE6_MODE	Current mode of slice 6	Disabled
15:14	SLICE7_MODE	Current mode of slice 7	Disabled
31:16	Reserved (zero)	-	-

Register [PTM_PARTITION_CONFIG.PTM_SLICE_MODE](#) layout

Field SLICE0_MODE

Current mode of slice 0.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE0_MODE](#) (S0) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE0_MODE values

Field SLICE1_MODE

Current mode of slice 1.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE1_MODE](#) (S1) is stored in bits [3:2] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE1_MODE values

Field SLICE2_MODE

Current mode of slice 2.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE2_MODE](#) (S2) is stored in bits [5:4] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE2_MODE values

Field SLICE3_MODE

Current mode of slice 3.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE3_MODE](#) (S3) is stored in bits [7:6] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE3_MODE values

Field SLICE4_MODE

Current mode of slice 4.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE4_MODE](#) (S4) is stored in bits [9:8] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE4_MODE values

Field SLICE5_MODE

Current mode of slice 5.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE5_MODE](#) (S5) is stored in bits [11:10] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE5_MODE values

Field SLICE6_MODE

Current mode of slice 6.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE6_MODE](#) (S6) is stored in bits [13:12] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE6_MODE values

Field SLICE7_MODE

Current mode of slice 7.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE.SLICE7_MODE](#) (S7) is stored in bits [15:14] and is a 2-bit enumeration of type PTM_SLICE_STATE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Transitioning	Slice is changing state

Field SLICE7_MODE values

PTM_SLICE_MODE_NEW (0x0018)

Set slice mode.

Sets the slice configuration. The values written to this register become effective when PTM_SLICE_MODE_ACK is set true following a write to PTM_SLICE_MODE_UPDATE. Fields for unassigned slices ignore writes and read zero. Register is read-only while PARTITION_CONTROL.PTM_PARTITION_STATE.LOCK is asserted.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
1:0	SLICE0_MODE	Intended mode of slice 0	Disabled
3:2	SLICE1_MODE	Intended mode of slice 1	Disabled
5:4	SLICE2_MODE	Intended mode of slice 2	Disabled
7:6	SLICE3_MODE	Intended mode of slice 3	Disabled
9:8	SLICE4_MODE	Intended mode of slice 4	Disabled
11:10	SLICE5_MODE	Intended mode of slice 5	Disabled
13:12	SLICE6_MODE	Intended mode of slice 6	Disabled
15:14	SLICE7_MODE	Intended mode of slice 7	Disabled
31:16	Reserved (zero)	-	-

Register [PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#) layout

Field SLICE0_MODE

Intended mode of slice 0.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW.SLICE0_MODE](#) (M0) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE0_MODE values

Field SLICE1_MODE

Intended mode of slice 1.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW.SLICE1_MODE](#) (M1) is stored in bits [3:2] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE1_MODE values

Field SLICE2_MODE

Intended mode of slice 2.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE2_MODE (M2) is stored in bits [5:4] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE2_MODE values

Field SLICE3_MODE

Intended mode of slice 3.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE3_MODE (M3) is stored in bits [7:6] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE3_MODE values

Field SLICE4_MODE

Intended mode of slice 4.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE4_MODE (M4) is stored in bits [9:8] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager

Value	Name	Meaning
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE4_MODE values

Field SLICE5_MODE

Intended mode of slice 5.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE5_MODE (M5) is stored in bits [11:10] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE5_MODE values

Field SLICE6_MODE

Intended mode of slice 6.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE6_MODE (M6) is stored in bits [13:12] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE6_MODE values

Field SLICE7_MODE

Intended mode of slice 7.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_NEW](#).SLICE7_MODE (M7) is stored in bits [15:14] and is a 2-bit enumeration of type PTM_SLICE_MODE_ENUM. Its default value is Disabled.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disabled	Slice is disabled
1	Manager	Slice is enabled and a manager
2	Subordinate	Slice is enabled and a subordinate
3	Reserved	-

Field SLICE7_MODE values

PTM_SLICE_MODE_UPDATE (0x001C)

Update slice mode.

Update internal state to the PTM_SLICE_MODE_NEW value. Writing to this register causes PTM_SLICE_MODE_ACK to go low and stay low until the new configuration has been updated. This register is automatically cleared when the update has completed. Register is read-only while PARTITION_CONTROL.PTM_PARTITION_STATE.LOCK is asserted.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	UPDATE	Set true to update mode	0
31:1	Reserved (zero)		-

Register [PTM_PARTITION_CONFIG.PTM_SLICE_MODE_UPDATE](#) layout

Field UPDATE

Set true to update mode.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_UPDATE.UPDATE](#) (UP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No update or update complete
1	Perform update

Field UPDATE values

PTM_SLICE_MODE_ACK (0x0020)

Acknowledge update of slice mode.

Software should poll this register after updating the configuration; when read as true software should set it false to complete the update handshake. If a slice is enabled but is already enabled in another partition the error flag is set. The error is cleared by setting the slices to a valid configuration and repeating the update.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	STATUS	Status of configuration update	0
1	ERROR	Error in slice configuration	0
31:2	Reserved (zero)	-	-

Register [PTM_PARTITION_CONFIG.PTM_SLICE_MODE_ACK](#) layout

Field STATUS

Status of configuration update.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_ACK.STATUS](#) (S) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Update ongoing or handshake complete
1	Update ready but handshake incomplete

Field STATUS values

Field ERROR

Error in slice configuration.

[PTM_PARTITION_CONFIG.PTM_SLICE_MODE_ACK.ERROR](#) (E) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Attempt to enable a slice that is enabled in another partition

Field ERROR values

C.1.7 Register sub-page PTM_PARTITION_CONTROL

Layout

Registers for controlling a partition.

Address	Name	Usage	Access
0x0000	PTM_ID	Partition Manager ID	ro
0x0004	PTM_UNIT_FEATURES	Partition features	ro
0x0008	PTM_SLICE_FEATURES	Slice features	ro
0x000C - 0x0010	PTM_SLICE_CORES	Cores per slice	ro
0x0014 - 0x003C	Reserved	-	-
0x0040	PTM_IRQ_RAWSTAT	Status of partition control error interrupts before masking	ro
0x0044	PTM_IRQ_CLEAR	Clear partition control error interrupts	rw
0x0048	PTM_IRQ_MASK	Enable and disable partition control error interrupts	rw
0x004C	PTM_IRQ_STATUS	Status of partition control error interrupts after masking	ro
0x0050	PTM_IRQ_INJECTION	Inject partition control interrupts (before masking)	rw
0x0054	PTM_ERROR_FINGER_PRINT	Information indicating the cause of an error reported by an interrupt	rw
0x0058	PTM_RESET_STATE	Reset state	ro
0x005C	PTM_RESET_SET	Set reset	rw
0x0060	PTM_PARTITION_STATE	Partition state	ro
0x0064	PTM_AW_SET	Set access window for this partition	rw
0x0068	PTM_BIST_SET	Set this partition into BIST mode	rw
0x006C	PTM_MTCRC_SET	Enable memory transaction checking for this partition	rw
0x0070 - 0x0FFC	Reserved		

0x1000 - 0x11FC	PTM_SLICE0_BIST	BIST control registers for slice 0	rw
0x1200 - 0x13FC	PTM_SLICE1_BIST	BIST control registers for slice 1	rw
0x1400 - 0x15FC	PTM_SLICE2_BIST	BIST control registers for slice 2	rw
0x1600 - 0x17FC	PTM_SLICE3_BIST	BIST control registers for slice 3	rw
0x1800 - 0x19FC	PTM_SLICE4_BIST	BIST control registers for slice 4	rw
0x1A00 - 0x1BFC	PTM_SLICE5_BIST	BIST control registers for slice 5	rw
0x1C00 - 0x1DFC	PTM_SLICE6_BIST	BIST control registers for slice 6	rw
0x1E00 - 0x1FFC	PTM_SLICE7_BIST	BIST control registers for slice 7	rw
0x2000 - 0xFFFFD	Reserved	-	-

List of registers

PTM_ID (0x0000)

Partition Manager ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_PARTITION_CONTROL.PTM_ID](#) layout

Field [VERSION_STATUS](#)

Contains the status of the GPU release.

[PTM_PARTITION_CONTROL.PTM_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field **VERSION_MINOR**

Contains the minor release number of the GPU (the P part of an RnP_n release number).

[PTM_PARTITION_CONTROL.PTM_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **VERSION_MAJOR**

Contains the major release number of the GPU (the R part of an RnP_n release number).

[PTM_PARTITION_CONTROL.PTM_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **PRODUCT_MAJOR**

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_PARTITION_CONTROL.PTM_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field **ARCH_REV**

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONTROL.PTM_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_MINOR

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONTROL.PTM_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field ARCH_MAJOR

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_PARTITION_CONTROL.PTM_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_UNIT_FEATURES (0x0004)

Partition features.

Identifies the partition manager features.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITIONS	Total number of partitions in the system	Implementation defined	1	4
7:4	Reserved (zero)	-	-	-	-
13:8	ACCESS_WINDOWS	Total number of access windows in the system	Implementation defined	1	16
31:14	Reserved (zero)	-	-	-	-

Register [PTM_PARTITION_CONTROL.PTM_UNIT_FEATURES](#) layout

Field PARTITIONS

Total number of partitions in the system.

[PTM_PARTITION_CONTROL.PTM_UNIT_FEATURES.PARTITIONS](#) (P) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 1 to 4 inclusive. Its default value is Implementation defined.

Field ACCESS_WINDOWS

Total number of access windows in the system.

[PTM_PARTITION_CONTROL.PTM_UNIT_FEATURES.ACCESS_WINDOWS](#) (W) is stored in bits [13:8] and is a 6-bit unsigned integer. Its value must lie in the range from 1 to 16 inclusive. Its default value is Implementation defined.

PTM_SLICE_FEATURES (0x0008)

Slice features.

Reports features of each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
7:0	CORE_MASK_STRIDE	Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice)	4
8	SLICE0_TILER	Type of tiler in slice 0	Implementation defined
9	SLICE1_TILER	Type of tiler in slice 1	Implementation defined
10	SLICE2_TILER	Type of tiler in slice 2	Implementation defined
11	SLICE3_TILER	Type of tiler in slice 3	Implementation defined
12	SLICE4_TILER	Type of tiler in slice 4	Implementation defined
13	SLICE5_TILER	Type of tiler in slice 5	Implementation defined
14	SLICE6_TILER	Type of tiler in slice 6	Implementation defined
15	SLICE7_TILER	Type of tiler in slice 7	Implementation defined
31:16	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES](#) layout

Field [CORE_MASK_STRIDE](#)

Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice).

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.CORE_MASK_STRIDE](#) (S) is stored in bits [7:0] and is a 8-bit unsigned integer. Its default value is 4.

Field [SLICE0_TILER](#)

Type of tiler in slice 0.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE0_TILER](#) (T0) is stored in bit [8] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE0_TILER values

Field SLICE1_TILER

Type of tiler in slice 1.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE1_TILER](#) (T1) is stored in bit [9] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE1_TILER values

Field SLICE2_TILER

Type of tiler in slice 2.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE2_TILER](#) (T2) is stored in bit [10] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE2_TILER values

Field SLICE3_TILER

Type of tiler in slice 3.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE3_TILER](#) (T3) is stored in bit [11] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE3_TILER values

Field SLICE4_TILER

Type of tiler in slice 4.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE4_TILER](#) (T4) is stored in bit [12] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE4_TILER values

Field SLICE5_TILER

Type of tiler in slice 5.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES.SLICE5_TILER](#) (T5) is stored in bit [13] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE5_TILER values

Field SLICE6_TILER

Type of tiler in slice 6.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES](#).SLICE6_TILER (T6) is stored in bit [14] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE6_TILER values

Field SLICE7_TILER

Type of tiler in slice 7.

[PTM_PARTITION_CONTROL.PTM_SLICE_FEATURES](#).SLICE7_TILER (T7) is stored in bit [15] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE7_TILER values

PTM_SLICE_CORES (0x000C - 0x0010)

Cores per slice.

Identifies the number of cores in each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
7:0	CORES0	Number of shader cores in slice 0	Implementation defined	0	3
15:8	CORES1	Number of shader cores in slice 1	Implementation defined	0	3
23:16	CORES2	Number of shader cores in slice 2	Implementation defined	0	3
31:24	CORES3	Number of shader cores in slice 3	Implementation defined	0	3
39:32	CORES4	Number of shader cores in slice 4	Implementation defined	0	3
47:40	CORES5	Number of shader cores in slice 5	Implementation defined	0	3
55:48	CORES6	Number of shader cores in slice 6	Implementation defined	0	3
63:56	CORES7	Number of shader cores in slice 7	Implementation defined	0	3

Register [PTM_PARTITION_CONTROL.PTM_SLICE_CORES](#) layout

Field CORES0

Number of shader cores in slice 0.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES0](#) (C0) is stored in bits [7:0] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES1

Number of shader cores in slice 1.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES1](#) (C1) is stored in bits [15:8] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES2

Number of shader cores in slice 2.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES2](#) (C2) is stored in bits [23:16] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES3

Number of shader cores in slice 3.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES3](#) (C3) is stored in bits [31:24] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES4

Number of shader cores in slice 4.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES4](#) (C4) is stored in bits [39:32] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES5

Number of shader cores in slice 5.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES5](#) (C5) is stored in bits [47:40] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES6

Number of shader cores in slice 6.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES6](#) (C6) is stored in bits [55:48] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES7

Number of shader cores in slice 7.

[PTM_PARTITION_CONTROL.PTM_SLICE_CORES.CORES7](#) (C7) is stored in bits [63:56] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

PTM_IRQ_RAWSTAT (0x0040)

Status of partition control error interrupts before masking.

Report of partition control errors before masking.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	INVALID_ACCESS	Attempt to access a locked register	0
1	INVALID_BIST	Attempt to access a locked BIST region	0
2	MTCRC	Memory transaction CRC error	0
7:3	Reserved (zero)	-	-
15:8	SLICE_BIST_DONE	BIST complete for slice	0
23:16	SLICE_BIST_ERROR	BIST error for slice	0
31:24	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT](#) layout

Field INVALID_ACCESS

Attempt to access a locked register.

[PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT.INVALID_ACCESS](#) (IA) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field [INVALID_ACCESS](#) values

Field INVALID_BIST

Attempt to access a locked BIST region.

[PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT.INVALID_BIST](#) (IB) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field [INVALID_BIST](#) values

Field MTCRC

Memory transaction CRC error.

[PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT.MTCRC](#) (MT) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field MTCRC values

Field SLICE_BIST_DONE

BIST complete for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT](#).SLICE_BIST_DONE is stored in bits [15:8] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Not complete
1	Complete

Field SLICE_BIST_DONE bit values.

Field SLICE_BIST_ERROR

BIST error for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_RAWSTAT](#).SLICE_BIST_ERROR is stored in bits [23:16] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST_ERROR bit values

PTM_IRQ_CLEAR (0x0044)

Clear partition control error interrupts.

Write '1' to clear the corresponding interrupt; this register always reads as zero.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	INVALID_ACCESS	Clear invalid access error	0
1	INVALID_BIST	Clear invalid BIST error	0
2	MTCRC	Clear memory transaction CRC error	0
7:3	Reserved (zero)	-	-
15:8	SLICE_BIST_DONE	Clear BIST done for slice	0
23:16	SLICE_BIST_ERROR	Clear BIST error for slice	0
31:24	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR](#) layout

Field INVALID_ACCESS

Clear invalid access error.

[PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR](#).INVALID_ACCESS (IA) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field INVALID_ACCESS values

Field INVALID_BIST

Clear invalid BIST error.

[PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR.INVALID_BIST](#) (IB) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field INVALID_BIST values

Field MTCRC

Clear memory transaction CRC error.

[PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR.MTCRC](#) (MT) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MTCRC values

Field SLICE_BIST_DONE

BIST complete for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR.SLICE_BIST_DONE](#) is stored in bits [15:8] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Do nothing
1	Clear BIST done

Field SLICE_BIST_DONE bit values.

Field SLICE_BIST_ERROR

BIST error for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_CLEAR.SLICE_BIST_ERROR](#) is stored in bits [23:16] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE_BIST_ERROR bit values

PTM_IRQ_MASK (0x0048)

Enable and disable partition control error interrupts.

Set to '1' to enable the corresponding interrupt.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	INVALID_ACCESS	Enable invalid access error	0
1	INVALID_BIST	Enable invalid BIST error	0
2	MTCRC	Enable memory transaction CRC error	0
7:3	Reserved (zero)	-	-
15:8	SLICE_BIST_DONE	Enable BIST complete for slice	0
23:16	SLICE_BIST_ERROR	Enable BIST error for slice	0
31:24	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL](#).PTM_IRQ_MASK layout

Field INVALID_ACCESS

Enable invalid access error.

[PTM_PARTITION_CONTROL](#).PTM_IRQ_MASK.INVALID_ACCESS (IA) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_ACCESS values

Field INVALID_BIST

Enable invalid BIST error.

[PTM_PARTITION_CONTROL](#).PTM_IRQ_MASK.INVALID_BIST (IB) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_BIST values

Field MTCRC

Enable memory transaction CRC error.

[PTM_PARTITION_CONTROL](#).PTM_IRQ_MASK.MTCRC (MT) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable

1	Enable
---	--------

Field MTCRC values

Field SLICE_BIST_DONE

BIST complete for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_MASK.SLICE_BIST_DONE](#) is stored in bits [15:8] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE_BIST_DONE bit values.

Field SLICE_BIST_ERROR

BIST error for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_MASK.SLICE_BIST_ERROR](#) is stored in bits [23:16] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE_BIST_ERROR bit values

PTM_IRQ_STATUS (0x004C)

Status of partition control error interrupts after masking.

Report of partition errors after masking; an interrupt is raised if any bit is '1'.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	INVALID_ACCESS	Invalid access (attempt to access a locked register)	0
1	INVALID_BIST	Attempt to access a disabled BIST region	0
2	MTCRC	Memory transaction CRC error	0
7:3	Reserved (zero)	-	-
15:8	SLICE_BIST_DONE	BIST complete for slice	0
23:16	SLICE_BIST_ERROR	BIST error for slice	0
31:24	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_IRQ_STATUS](#) layout

Field INVALID_ACCESS

Invalid access (attempt to access a locked register).

[PTM_PARTITION_CONTROL.PTM_IRQ_STATUS.INVALID_ACCESS](#) (IA) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_ACCESS values

Field INVALID_BIST

Attempt to access a disabled BIST region.

[PTM_PARTITION_CONTROL.PTM_IRQ_STATUS.INVALID_BIST](#) (IB) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BIST values

Field MTCRC

Memory transaction CRC error.

[PTM_PARTITION_CONTROL.PTM_IRQ_STATUS.MTCRC](#) (MT) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field MTCRC values

Field SLICE_BIST_DONE

BIST complete for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_STATUS.SLICE_BIST_DONE](#) is stored in bits [15:8] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Not complete
1	Complete

Field SLICE_BIST_DONE bit values.

Field SLICE_BIST_ERROR

BIST error for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_STATUS.SLICE_BIST_ERROR](#) is stored in bits [23:16] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field SLICE_BIST_ERROR bit values

PTM_IRQ_INJECTION (0x0050)

Inject partition control interrupts (before masking).

Inject interrupts to test interrupt handlers are correctly configured.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	INVALID_ACCESS	Invalid access (attempt to access a locked register)	0
1	INVALID_BIST	Attempt to access a disabled BIST region	0
2	MTCRC	Memory transaction CRC error	0
7:3	Reserved (zero)	-	-
15:8	SLICE_BIST_DONE	BIST complete for slice	0
23:16	SLICE_BIST_ERROR	BIST error for slice	0
31:24	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL](#).PTM_IRQ_INJECTION layout

Field INVALID_ACCESS

Invalid access (attempt to access a locked register).

[PTM_PARTITION_CONTROL](#).PTM_IRQ_INJECTION.INVALID_ACCESS (IA) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_ACCESS values

Field INVALID_BIST

Attempt to access a disabled BIST region.

[PTM_PARTITION_CONTROL](#).PTM_IRQ_INJECTION.INVALID_BIST (IB) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BIST values

Field MTCRC

Memory transaction CRC error.

[PTM_PARTITION_CONTROL.PTM_IRQ_INJECTION](#).MTCRC (MT) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field MTCRC values

Field SLICE_BIST_DONE

BIST complete for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_INJECTION](#).SLICE_BIST_DONE is stored in bits [15:8] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	Not complete
1	Complete

Field SLICE_BIST_DONE bit values.

Field SLICE_BIST_ERROR

BIST error for slice.

[PTM_PARTITION_CONTROL.PTM_IRQ_INJECTION](#).SLICE_BIST_ERROR is stored in bits [23:16] and is a 8-bit bitset. Its default value is 0.

The nth bit in the bitset (n=0 to 7) has the following meaning:

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST_ERROR bit values

PTM_ERROR_FINGER_PRINT (0x0054)

Information indicating the cause of an error reported by an interrupt.

Used to indicate the cause of an error reported by an interrupt. The state of this register is preserved until the valid bit is cleared; errors arising while the valid bit is set do not record their cause.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	VALID	Register holds valid data	0
31:1	FINGER_PRINT	Indicates cause of error including source (format)	0

Register [PTM_PARTITION_CONTROL.PTM_ERROR_FINGER_PRINT](#) layout

Field VALID

Register holds valid data.

[PTM_PARTITION_CONTROL.PTM_ERROR_FINGER_PRINT.VALID](#) (V) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not valid
1	Valid

Field VALID values

Field FINGER_PRINT

Indicates cause of error including source (format).

[PTM_PARTITION_CONTROL.PTM_ERROR_FINGER_PRINT.FINGER_PRINT](#) (F) is stored in bits [31:1] and is a 31-bit unsigned integer. Its default value is 0.

PTM_RESET_STATE (0x0058)

Reset state.

Indicates the state of the partition reset.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	RESET_STATE	Reset state of partition
31:1	Reserved (zero)	-

Register [PTM_PARTITION_CONTROL.PTM_RESET_STATE](#) layout

Field RESET_STATE

Reset state of partition.

[PTM_PARTITION_CONTROL.PTM_RESET_STATE.RESET_STATE](#) (R) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
0	Not reset
1	Reset

Field RESET_STATE values

PTM_RESET_SET (0x005C)

Set reset.

To reset a partition set this register and poll PTM_RESET_STATE until it shows the same state. Then return this register to 'no reset' and wait for status to do the same. This reset is used to unlock the partition control and configuration registers. Once the reset has been initiated it cannot be canceled.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	RESET	Set reset of partition	0
31:1	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL](#).PTM_RESET_SET layout

Field RESET

Set reset of partition.

[PTM_PARTITION_CONTROL](#).PTM_RESET_SET.RESET (R) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not reset
1	Reset

Field RESET values

PTM_PARTITION_STATE (0x0060)

Partition state.

State of this partition and the access window using it.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
1:0	LOCK	Lock state of partition control registers	UNLOCKED
2	AW_READY	State of access window	0
3	AW_ERROR	Error when enabling an access window; cleared by partition reset	0
4	BIST_ERROR	Error when enabling BIST; cleared by partition reset	0
31:5	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL](#).PTM_PARTITION_STATE layout

Field LOCK

Lock state of partition control registers.

[PTM_PARTITION_CONTROL](#).PTM_PARTITION_STATE.LOCK (A) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_LOCK_ENUM. Its default value is UNLOCKED.

The field can contain the following values:

Value	Name	Meaning
0 (default)	UNLOCKED	Not locked
1	AW_LOCK	Locked by writing to PTM_AW_SET
2	BIST_LOCK	Locked by writing to PTM_BIST_SET
3	Reserved	-

Field LOCK values

Field AW_READY

State of access window.

[PTM_PARTITION_CONTROL.PTM_PARTITION_STATE.AW_READY](#) (R) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Access window is not ready for use
1	Access window is ready for use

Field AW_READY values

Field AW_ERROR

Error when enabling an access window; cleared by partition reset.

[PTM_PARTITION_CONTROL.PTM_PARTITION_STATE.AW_ERROR](#) (E) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error (attempt to enable a window already enabled on another partition or more than one window on this partition)

Field AW_ERROR values

Field BIST_ERROR

Error when enabling an access window; cleared by partition reset.

[PTM_PARTITION_CONTROL.PTM_PARTITION_STATE.BIST_ERROR](#) (B) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error (attempt to modify PTM_BIST_SET failed)

Field BIST_ERROR values

PTM_AW_SET (0x0064)

Set access window for this partition.

Enable an access window for this partition. No more than one bit may be set at any time. Setting any bit in this register locks partition configuration and control registers. See text for full sequence. This register is reset by PTM_RESET_SET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	AW0	Enable window 0 for this partition	0
1	AW1	Enable window 1 for this partition	0
2	AW2	Enable window 2 for this partition	0
3	AW3	Enable window 3 for this partition	0
4	AW4	Enable window 4 for this partition	0

Bits	Name	Usage	Default
5	AW5	Enable window 5 for this partition	0
6	AW6	Enable window 6 for this partition	0
7	AW7	Enable window 7 for this partition	0
8	AW8	Enable window 8 for this partition	0
9	AW9	Enable window 9 for this partition	0
10	AW10	Enable window 10 for this partition	0
11	AW11	Enable window 11 for this partition	0
12	AW12	Enable window 12 for this partition	0
13	AW13	Enable window 13 for this partition	0
14	AW14	Enable window 14 for this partition	0
15	AW15	Enable window 15 for this partition	0
31:16	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_AW_SET](#) layout

Field AW0

Enable window 0 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET.AW0](#) (W0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW0 values

Field AW1

Enable window 1 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET.AW1](#) (W1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW1 values

Field AW2

Enable window 2 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET.AW2](#) (W2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW2 values

Field AW3

Enable window 3 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW3 (W3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW3 values

Field AW4

Enable window 4 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW4 (W4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW4 values

Field AW5

Enable window 5 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW5 (W5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW5 values

Field AW6

Enable window 6 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW6 (W6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW6 values

Field AW7

Enable window 7 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW7 (W7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Not enabled
1	Enabled

Field AW7 values

Field AW8

Enable window 8 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW8 (W8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW8 values

Field AW9

Enable window 9 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW9 (W9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW9 values

Field AW10

Enable window 10 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW10 (W10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW10 values

Field AW11

Enable window 11 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW11 (W11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW11 values

Field AW12

Enable window 12 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW12 (W12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW12 values

Field AW13

Enable window 13 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW13 (W13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW13 values

Field AW14

Enable window 14 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW14 (W14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW14 values

Field AW15

Enable window 15 for this partition.

[PTM_PARTITION_CONTROL.PTM_AW_SET](#).AW15 (W15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field AW15 values

PTM_BIST_SET (0x0068)

Set this partition into BIST mode.

Allow access to BIST controllers for each slice in this partition (set to primary or secondary). Setting this register locks partition configuration and control registers. See text for details. This register is read-only while PTM_PARTITION_STATE.LOCK is set and is reset by PTM_RESET_SET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	BIST	Enable BIST for this partition	0
31:1	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_BIST_SET](#) layout

Field BIST

Enable BIST for this partition.

[PTM_PARTITION_CONTROL.PTM_BIST_SET](#).BIST (B) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field BIST values

PTM_MTCRC_SET (0x006C)

Enable memory transaction checking for this partition.

Enable memory transaction CRC checking for this partition

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MTCRC	Enable memory transaction CRC for this partition	0
31:1	Reserved (zero)	-	-

Register [PTM_PARTITION_CONTROL.PTM_MTCRC_SET](#) layout

Field MTCRC

Enable memory transaction CRC for this partition.

[PTM_PARTITION_CONTROL.PTM_MTCRC_SET](#).MTCRC (M) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not enabled
1	Enabled

Field MTCRC values

PTM_SLICE0_BIST (0x1000 - 0x11FC)

BIST control registers for slice 0.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE1_BIST (0x1200 - 0x13FC)

BIST control registers for slice 1.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE2_BIST (0x1400 - 0x15FC)

BIST control registers for slice 2.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE3_BIST (0x1600 - 0x17FC)

BIST control registers for slice 3.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE4_BIST (0x1800 - 0x19FC)

BIST control registers for slice 4.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE5_BIST (0x1A00 - 0x1BFC)

BIST control registers for slice 5.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE6_BIST (0x1C00 - 0x1DFC)

BIST control registers for slice 6.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_SLICE7_BIST (0x1E00 - 0x1FFC)

BIST control registers for slice 7.

BIST control registers. Details on the [PTM_BIST_CONTROL](#) sub-page.

Access to this Register is restricted to mode [rw](#).

C.1.8 Register sub-page PTM_RESOURCE_GROUP

Register sub-page PTM_RESOURCE_GROUP

Layout

Resource group.

Address	Name	Usage	Access
0x0000	PTM_ID	Partition Manager and GPU ID	ro
0x0004	PTM_UNIT_FEATURES	Partition features	ro
0x0008	PTM_SLICE_FEATURES	Slice features	ro
0x000C - 0x0010	PTM_SLICE_CORES	Cores per slice	ro
0x0014	PTM_SLICE_MASK	Mask of slices assigned to this group	ro
0x0018	PTM_PARTITION_MASK	Mask of partitions assigned to this group	ro
0x001C	PTM_AW_MASK	Mask of access windows assigned to this group	ro
0x0020 - 0x007C	Reserved	-	-
0x0080	PTM_IRQ_RAWSTAT	Status of incoming message interrupt before masking	ro
0x0084	PTM_IRQ_CLEAR	Clear incoming message	rw
0x0088	PTM_IRQ_MASK	Enable and disable the message interrupt	rw
0x008C	PTM_IRQ_STATUS	Status of the message after masking	ro
0x0090	PTM_IRQ_INJECTION	Inject interrupt before masking	rw
0x0094	PTM_WATCHDOG	Watchdog register	rw
0x0098	PTM_SLICE_POWER_STATE	Power state of slices	ro
0x009C	PTM_SLICE_POWER_SET	Set slice power mode	rw
0x00A0	PTM_SLICE_CLOCK_STATE	Clock state of slices	ro
0x00A4	PTM_SLICE_CLOCK_SET	Set slice clock mode	rw
0x00A8	PTM_SLICE_RESET_STATE	Reset state of slices	ro
0x00AC	PTM_SLICE_RESET_SET	Set slice reset	rw
0x00B0 - 0x00FC	Reserved	-	-
0x0100 - 0x011C	PTM_AW0_MESSAGE	Access window 0 message registers	rw
0x0120 - 0x013C	PTM_AW1_MESSAGE	Access window 1 message registers	rw
0x0140 - 0x015C	PTM_AW2_MESSAGE	Access window 2 message registers	rw
0x0160 - 0x017C	PTM_AW3_MESSAGE	Access window 3 message registers	rw
0x0180 - 0x019C	PTM_AW4_MESSAGE	Access window 4 message registers	rw
0x01A0 - 0x01BC	PTM_AW5_MESSAGE	Access window 5 message registers	rw
0x01C0 - 0x01DC	PTM_AW6_MESSAGE	Access window 6 message registers	rw
0x01E0 - 0x01FC	PTM_AW7_MESSAGE	Access window 7 message registers	rw
0x0200 - 0x021C	PTM_AW8_MESSAGE	Access window 8 message registers	rw
0x0220 - 0x023C	PTM_AW9_MESSAGE	Access window 9 message registers	rw
0x0240 - 0x025C	PTM_AW10_MESSAGE	Access window 10 message registers	rw
0x0260 - 0x027C	PTM_AW11_MESSAGE	Access window 11 message registers	rw

0x0280 - 0x029C	PTM_AW12_MESSAGE	Access window 12 message registers	rw
0x02A0 - 0x02BC	PTM_AW13_MESSAGE	Access window 13 message registers	rw
0x02C0 - 0x02DC	PTM_AW14_MESSAGE	Access window 14 message registers	rw
0x02E0 - 0x02FC	PTM_AW15_MESSAGE	Access window 15 message registers	rw
0x0300 - 0xFFFFD	Reserved	-	-

List of registers

PTM_ID (0x0000)

Partition Manager and GPU ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_RESOURCE_GROUP.PTM_ID](#) layout

Field [VERSION_STATUS](#)

Contains the status of the GPU release.

[PTM_RESOURCE_GROUP.PTM_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field VERSION_MINOR

Contains the minor release number of the GPU (the P part of an RnPn release number).

[PTM_RESOURCE_GROUP.PTM_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field VERSION_MAJOR

Contains the major release number of the GPU (the R part of an RnPn release number).

[PTM_RESOURCE_GROUP.PTM_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field PRODUCT_MAJOR

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_RESOURCE_GROUP.PTM_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_REV

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_RESOURCE_GROUP.PTM_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field ARCH_MINOR

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_RESOURCE_GROUP.PTM_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field ARCH_MAJOR

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_RESOURCE_GROUP.PTM_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_UNIT_FEATURES (0x0004)

Partition features.

Identifies the partition manager features.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITIONS	Total number of partitions in the system	Implementation defined	1	4
7:4	Reserved (zero)	-	-	-	-
13:8	ACCESS_WINDOWS	Total number of access windows in the system	Implementation defined	1	16
31:14	Reserved (zero)	-	-	-	-

Register [PTM_RESOURCE_GROUP.PTM_UNIT_FEATURES](#) layout

Field PARTITIONS

Total number of partitions in the system.

[PTM_RESOURCE_GROUP.PTM_UNIT_FEATURES.PARTITIONS](#) (P) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 1 to 4 inclusive. Its default value is Implementation defined.

Field ACCESS_WINDOWS

Total number of access windows in the system.

[PTM_RESOURCE_GROUP.PTM_UNIT_FEATURES.ACCESS_WINDOWS](#) (W) is stored in bits [13:8] and is a 6-bit unsigned integer. Its value must lie in the range from 1 to 16 inclusive. Its default value is Implementation defined.

PTM_SLICE_FEATURES (0x0008)

Slice features.

Reports features of each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
7:0	CORE_MASK_STRIDE	Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice)	4
8	SLICE0_TILER	Type of tiler in slice 0	Implementation defined
9	SLICE1_TILER	Type of tiler in slice 1	Implementation defined
10	SLICE2_TILER	Type of tiler in slice 2	Implementation defined
11	SLICE3_TILER	Type of tiler in slice 3	Implementation defined
12	SLICE4_TILER	Type of tiler in slice 4	Implementation defined
13	SLICE5_TILER	Type of tiler in slice 5	Implementation defined
14	SLICE6_TILER	Type of tiler in slice 6	Implementation defined
15	SLICE7_TILER	Type of tiler in slice 7	Implementation defined
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES](#) layout

Field CORE_MASK_STRIDE

Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice).

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.CORE_MASK_STRIDE](#) (S) is stored in bits [7:0] and is a 8-bit unsigned integer. Its default value is 4.

Field SLICE0_TILER

Type of tiler in slice 0.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE0_TILER](#) (T0) is stored in bit [8] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field [SLICE0_TILER](#) values

Field SLICE1_TILER

Type of tiler in slice 1.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE1_TILER](#) (T1) is stored in bit [9] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE1_TILER values

Field SLICE2_TILER

Type of tiler in slice 2.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE2_TILER](#) (T2) is stored in bit [10] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE2_TILER values

Field SLICE3_TILER

Type of tiler in slice 3.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE3_TILER](#) (T3) is stored in bit [11] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE3_TILER values

Field SLICE4_TILER

Type of tiler in slice 4.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE4_TILER](#) (T4) is stored in bit [12] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE4_TILER values

Field SLICE5_TILER

Type of tiler in slice 5.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE5_TILER](#) (T5) is stored in bit [13] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE5_TILER values

Field SLICE6_TILER

Type of tiler in slice 6.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE6_TILER](#) (T6) is stored in bit [14] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE6_TILER values

Field SLICE7_TILER

Type of tiler in slice 7.

[PTM_RESOURCE_GROUP.PTM_SLICE_FEATURES.SLICE7_TILER](#) (T7) is stored in bit [15] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE7_TILER values

PTM_SLICE_CORES (0x000C - 0x0010)

Cores per slice.

Identifies the number of cores in each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
7:0	CORES0	Number of shader cores in slice 0	Implementation defined	0	3
15:8	CORES1	Number of shader cores in slice 1	Implementation defined	0	3
23:16	CORES2	Number of shader cores in slice 2	Implementation defined	0	3
31:24	CORES3	Number of shader cores in slice 3	Implementation defined	0	3
39:32	CORES4	Number of shader cores in slice 4	Implementation defined	0	3
47:40	CORES5	Number of shader cores in slice 5	Implementation defined	0	3
55:48	CORES6	Number of shader cores in slice 6	Implementation defined	0	3
63:56	CORES7	Number of shader cores in slice 7	Implementation defined	0	3

Register [PTM_RESOURCE_GROUP.PTM_SLICE_CORES](#) layout

Field CORES0

Number of shader cores in slice 0.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES0](#) (C0) is stored in bits [7:0] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES1

Number of shader cores in slice 1.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES1](#) (C1) is stored in bits [15:8] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES2

Number of shader cores in slice 2.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES2](#) (C2) is stored in bits [23:16] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES3

Number of shader cores in slice 3.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES3](#) (C3) is stored in bits [31:24] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES4

Number of shader cores in slice 4.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES4](#) (C4) is stored in bits [39:32] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES5

Number of shader cores in slice 5.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES5](#) (C5) is stored in bits [47:40] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES6

Number of shader cores in slice 6.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES6](#) (C6) is stored in bits [55:48] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES7

Number of shader cores in slice 7.

[PTM_RESOURCE_GROUP.PTM_SLICE_CORES.CORES7](#) (C7) is stored in bits [63:56] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

PTM_SLICE_MASK (0x0014)

Mask of slices assigned to this group.

Mask of slices assigned to a group

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	SLICE0	Presence of slice 0 in this group
1	SLICE1	Presence of slice 1 in this group
2	SLICE2	Presence of slice 2 in this group
3	SLICE3	Presence of slice 3 in this group
4	SLICE4	Presence of slice 4 in this group
5	SLICE5	Presence of slice 5 in this group
6	SLICE6	Presence of slice 6 in this group
7	SLICE7	Presence of slice 7 in this group
31:8	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_MASK](#) layout

Field SLICE0

Presence of slice 0 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK](#).SLICE0 (S0) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE0 values

Field SLICE1

Presence of slice 1 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK](#).SLICE1 (S1) is stored in bit [1] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE1 values

Field SLICE2

Presence of slice 2 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK](#).SLICE2 (S2) is stored in bit [2] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE2 values

Field SLICE3

Presence of slice 3 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK.SLICE3](#) (S3) is stored in bit [3] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE3 values

Field SLICE4

Presence of slice 4 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK.SLICE4](#) (S4) is stored in bit [4] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE4 values

Field SLICE5

Presence of slice 5 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK.SLICE5](#) (S5) is stored in bit [5] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE5 values

Field SLICE6

Presence of slice 6 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK.SLICE6](#) (S6) is stored in bit [6] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE6 values

Field SLICE7

Presence of slice 7 in this group.

[PTM_RESOURCE_GROUP.PTM_SLICE_MASK.SLICE7](#) (S7) is stored in bit [7] and is a 1-bit flag.

Value	Meaning
0	Slice not present
1	Slice present

Field SLICE7 values

PTM_PARTITION_MASK (0x0018)

Mask of partitions assigned to this group.

Mask of partitions assigned to group

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	PARTITION0	Presence of partition 0 in this group
1	PARTITION1	Presence of partition 1 in this group
2	PARTITION2	Presence of partition 2 in this group
3	PARTITION3	Presence of partition 3 in this group
31:4	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP](#).PTM_PARTITION_MASK layout

Field PARTITION0

Presence of partition 0 in this group.

[PTM_RESOURCE_GROUP](#).PTM_PARTITION_MASK.PARTITION0 (P0) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
0	Partition not present
1	Partition present

Field PARTITION0 values

Field PARTITION1

Presence of partition 1 in this group.

[PTM_RESOURCE_GROUP](#).PTM_PARTITION_MASK.PARTITION1 (P1) is stored in bit [1] and is a 1-bit flag.

Value	Meaning
0	Partition not present
1	Partition present

Field PARTITION1 values

Field PARTITION2

Presence of partition 2 in this group.

[PTM_RESOURCE_GROUP](#).PTM_PARTITION_MASK.PARTITION2 (P2) is stored in bit [2] and is a 1-bit flag.

Value	Meaning
0	Partition not present
1	Partition present

Field PARTITION2 values

Field PARTITION3

Presence of partition 3 in this group.

[PTM_RESOURCE_GROUP.PTM_PARTITION_MASK](#).PARTITION3 (P3) is stored in bit [3] and is a 1-bit flag.

Value	Meaning
0	Partition not present
1	Partition present

Field PARTITION3 values

PTM_AW_MASK (0x001C)

Mask of access windows assigned to this group.

Mask of access windows assigned to a group

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	ACCESS_WINDOW0	Presence of access window 0 in this group
1	ACCESS_WINDOW1	Presence of access window 1 in this group
2	ACCESS_WINDOW2	Presence of access window 2 in this group
3	ACCESS_WINDOW3	Presence of access window 3 in this group
4	ACCESS_WINDOW4	Presence of access window 4 in this group
5	ACCESS_WINDOW5	Presence of access window 5 in this group
6	ACCESS_WINDOW6	Presence of access window 6 in this group
7	ACCESS_WINDOW7	Presence of access window 7 in this group
8	ACCESS_WINDOW8	Presence of access window 8 in this group
9	ACCESS_WINDOW9	Presence of access window 9 in this group
10	ACCESS_WINDOW10	Presence of access window 10 in this group
11	ACCESS_WINDOW11	Presence of access window 11 in this group
12	ACCESS_WINDOW12	Presence of access window 12 in this group
13	ACCESS_WINDOW13	Presence of access window 13 in this group
14	ACCESS_WINDOW14	Presence of access window 14 in this group
15	ACCESS_WINDOW15	Presence of access window 15 in this group
31:16	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP.PTM_AW_MASK](#) layout

Field ACCESS_WINDOW0

Presence of access window 0 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK](#).ACCESS_WINDOW0 (AW0) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
-------	---------

0	Access window not present
1	Access window present

Field ACCESS_WINDOW0 values

Field ACCESS_WINDOW1

Presence of access window 1 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW1](#) (AW1) is stored in bit [1] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW1 values

Field ACCESS_WINDOW2

Presence of access window 2 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW2](#) (AW2) is stored in bit [2] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW2 values

Field ACCESS_WINDOW3

Presence of access window 3 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW3](#) (AW3) is stored in bit [3] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW3 values

Field ACCESS_WINDOW4

Presence of access window 4 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW4](#) (AW4) is stored in bit [4] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW4 values

Field ACCESS_WINDOW5

Presence of access window 5 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW5](#) (AW5) is stored in bit [5] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW5 values

Field ACCESS_WINDOW6

Presence of access window 6 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW6](#) (AW6) is stored in bit [6] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW6 values

Field ACCESS_WINDOW7

Presence of access window 7 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW7](#) (AW7) is stored in bit [7] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW7 values

Field ACCESS_WINDOW8

Presence of access window 8 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW8](#) (AW8) is stored in bit [8] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW8 values

Field ACCESS_WINDOW9

Presence of access window 9 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW9](#) (AW9) is stored in bit [9] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW9 values

Field ACCESS_WINDOW10

Presence of access window 10 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW10](#) (AW10) is stored in bit [10] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW10 values

Field ACCESS_WINDOW11

Presence of access window 11 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW11](#) (AW11) is stored in bit [11] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW11 values

Field ACCESS_WINDOW12

Presence of access window 12 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW12](#) (AW12) is stored in bit [12] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW12 values

Field ACCESS_WINDOW13

Presence of access window 13 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW13](#) (AW13) is stored in bit [13] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW13 values

Field ACCESS_WINDOW14

Presence of access window 14 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW14](#) (AW14) is stored in bit [14] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW14 values

Field ACCESS_WINDOW15

Presence of access window 15 in this group.

[PTM_RESOURCE_GROUP.PTM_AW_MASK.ACCESS_WINDOW15](#) (AW15) is stored in bit [15] and is a 1-bit flag.

Value	Meaning
0	Access window not present
1	Access window present

Field ACCESS_WINDOW15 values

PTM_IRQ_RAWSTAT (0x0080)

Status of incoming message interrupt before masking.

Interrupt status before masking.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	MESSAGE0	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
1	MESSAGE1	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
2	MESSAGE2	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
3	MESSAGE3	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
4	MESSAGE4	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
5	MESSAGE5	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
6	MESSAGE6	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
7	MESSAGE7	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
8	MESSAGE8	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
9	MESSAGE9	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
10	MESSAGE10	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0

Bits	Name	Usage	Default
11	MESSAGE11	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
12	MESSAGE12	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
13	MESSAGE13	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
14	MESSAGE14	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
15	MESSAGE15	Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT](#) layout

Field MESSAGE0

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT](#).MESSAGE0 (M0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE0 values

Field MESSAGE1

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT](#).MESSAGE1 (M1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE1 values

Field MESSAGE2

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT](#).MESSAGE2 (M2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE2 values

Field MESSAGE3

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE3](#) (M3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE3 values

Field MESSAGE4

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE4](#) (M4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE4 values

Field MESSAGE5

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE5](#) (M5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE5 values

Field MESSAGE6

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE6](#) (M6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE6 values

Field MESSAGE7

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE7](#) (M7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE7 values

Field MESSAGE8

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE8](#) (M8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE8 values

Field MESSAGE9

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE9](#) (M9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE9 values

Field MESSAGE10

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE10](#) (M10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE10 values

Field MESSAGE11

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE11](#) (M11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE11 values

Field MESSAGE12

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE12](#) (M12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE12 values

Field MESSAGE13

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE13](#) (M13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE13 values

Field MESSAGE14

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE14](#) (M14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE14 values

Field MESSAGE15

Set on receipt of a message; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_RAWSTAT.MESSAGE15](#) (M15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No message
1	Message

Field MESSAGE15 values

PTM_IRQ_CLEAR (0x0084)

Clear incoming message.

Write '1' to clear the status; this register always reads as zero.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE0	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
1	MESSAGE1	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
2	MESSAGE2	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
3	MESSAGE3	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
4	MESSAGE4	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
5	MESSAGE5	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
6	MESSAGE6	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
7	MESSAGE7	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
8	MESSAGE8	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
9	MESSAGE9	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
10	MESSAGE10	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
11	MESSAGE11	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
12	MESSAGE12	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
13	MESSAGE13	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0

Bits	Name	Usage	Default
14	MESSAGE14	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
15	MESSAGE15	Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR](#) layout

Field MESSAGE0

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE0](#) (M0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE0 values

Field MESSAGE1

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE1](#) (M1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE1 values

Field MESSAGE2

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE2](#) (M2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE2 values

Field MESSAGE3

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE3](#) (M3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE3 values

Field MESSAGE4

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE4](#) (M4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE4 values

Field MESSAGE5

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE5](#) (M5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE5 values

Field MESSAGE6

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE6](#) (M6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE6 values

Field MESSAGE7

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE7](#) (M7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE7 values

Field MESSAGE8

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE8](#) (M8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE8 values

Field MESSAGE9

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE9](#) (M9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE9 values

Field MESSAGE10

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE10](#) (M10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE10 values

Field MESSAGE11

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE11](#) (M11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing

1	Clear error
---	-------------

Field MESSAGE11 values

Field MESSAGE12

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE12](#) (M12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE12 values

Field MESSAGE13

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE13](#) (M13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE13 values

Field MESSAGE14

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE14](#) (M14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE14 values

Field MESSAGE15

Clear message status; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_CLEAR.MESSAGE15](#) (M15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field MESSAGE15 values

PTM_IRQ_MASK (0x0088)

Enable and disable the message interrupt.

Set to '1' to enable the interrupt.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE0	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
1	MESSAGE1	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
2	MESSAGE2	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
3	MESSAGE3	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
4	MESSAGE4	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
5	MESSAGE5	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
6	MESSAGE6	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
7	MESSAGE7	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
8	MESSAGE8	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
9	MESSAGE9	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
10	MESSAGE10	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
11	MESSAGE11	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
12	MESSAGE12	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
13	MESSAGE13	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
14	MESSAGE14	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
15	MESSAGE15	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_IRQ_MASK](#) layout

Field MESSAGE0

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE0](#) (M0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE0 values

Field MESSAGE1

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE1](#) (M1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE1 values

Field MESSAGE2

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE2](#) (M2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE2 values

Field MESSAGE3

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE3](#) (M3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE3 values

Field MESSAGE4

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE4](#) (M4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE4 values

Field MESSAGE5

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE5](#) (M5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE5 values

Field MESSAGE6

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE6](#) (M6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE6 values

Field MESSAGE7

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE7](#) (M7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE7 values

Field MESSAGE8

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE8](#) (M8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE8 values

Field MESSAGE9

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE9](#) (M9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE9 values

Field MESSAGE10

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE10](#) (M10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE10 values

Field MESSAGE11

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE11](#) (M11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE11 values

Field MESSAGE12

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE12](#) (M12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE12 values

Field MESSAGE13

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE13](#) (M13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE13 values

Field MESSAGE14

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE14](#) (M14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE14 values

Field MESSAGE15

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_MASK.MESSAGE15](#) (M15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE15 values

PTM_IRQ_STATUS (0x008C)

Status of the message after masking.

Report of interrupt status after masking; an interrupt is raised if the bit is '1'.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	MESSAGE0	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
1	MESSAGE1	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
2	MESSAGE2	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
3	MESSAGE3	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
4	MESSAGE4	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
5	MESSAGE5	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
6	MESSAGE6	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
7	MESSAGE7	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
8	MESSAGE8	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
9	MESSAGE9	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
10	MESSAGE10	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
11	MESSAGE11	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
12	MESSAGE12	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
13	MESSAGE13	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
14	MESSAGE14	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
15	MESSAGE15	Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero	0
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_IRQ_STATUS](#) layout

Field **MESSAGE0**

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE0](#) (M0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Disable
1	Enable

Field MESSAGE0 values

Field MESSAGE1

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE1](#) (M1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE1 values

Field MESSAGE2

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE2](#) (M2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE2 values

Field MESSAGE3

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE3](#) (M3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE3 values

Field MESSAGE4

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE4](#) (M4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE4 values

Field MESSAGE5

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE5](#) (M5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE5 values

Field MESSAGE6

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE6](#) (M6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE6 values

Field MESSAGE7

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE7](#) (M7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE7 values

Field MESSAGE8

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE8](#) (M8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE8 values

Field MESSAGE9

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE9](#) (M9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE9 values

Field MESSAGE10

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE10](#) (M10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE10 values

Field MESSAGE11

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE11](#) (M11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE11 values

Field MESSAGE12

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE12](#) (M12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE12 values

Field MESSAGE13

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE13](#) (M13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE13 values

Field MESSAGE14

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE14](#) (M14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE14 values

Field MESSAGE15

Status of interrupt; if the access window corresponding to this message is not assigned to this group this bit reads as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_STATUS.MESSAGE15](#) (M15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE15 values

PTM_IRQ_INJECTION (0x0090)

Inject interrupt before masking.

Inject interrupts to test interrupt handlers are correctly configured.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	MESSAGE0	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
1	MESSAGE1	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0

Bits	Name	Usage	Default
2	MESSAGE2	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
3	MESSAGE3	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
4	MESSAGE4	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
5	MESSAGE5	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
6	MESSAGE6	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
7	MESSAGE7	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
8	MESSAGE8	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
9	MESSAGE9	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
10	MESSAGE10	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
11	MESSAGE11	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
12	MESSAGE12	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
13	MESSAGE13	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
14	MESSAGE14	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
15	MESSAGE15	Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero	0
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#) layout

Field MESSAGE0

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE0](#) (M0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE0 values

Field MESSAGE1

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE1](#) (M1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE1 values

Field MESSAGE2

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE2](#) (M2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE2 values

Field MESSAGE3

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE3](#) (M3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE3 values

Field MESSAGE4

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE4](#) (M4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE4 values

Field MESSAGE5

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION.MESSAGE5](#) (M5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE5 values

Field MESSAGE6

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE6 (M6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE6 values

Field MESSAGE7

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE7 (M7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE7 values

Field MESSAGE8

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE8 (M8) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE8 values

Field MESSAGE9

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE9 (M9) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable

1	Enable
---	--------

Field MESSAGE9 values

Field MESSAGE10

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE10 (M10) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE10 values

Field MESSAGE11

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE11 (M11) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE11 values

Field MESSAGE12

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE12 (M12) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE12 values

Field MESSAGE13

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE13 (M13) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE13 values

Field MESSAGE14

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE14 (M14) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE14 values

Field MESSAGE15

Enable message interrupt; if the access window corresponding to this message is not assigned to this group this bit is read-only as zero.

[PTM_RESOURCE_GROUP.PTM_IRQ_INJECTION](#).MESSAGE15 (M15) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field MESSAGE15 values

PTM_WATCHDOG (0x0094)

Watchdog register.

Value loaded into decrementing counter and an error raised when the counter transitions from one to zero. The error is reported through the system error interrupt (see the [PTM_SYSTEM](#) sub-page).

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage
15:0	TIMEOUT	Timeout value in multiples of 65536 GPU clocks
31:16	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP.PTM_WATCHDOG](#) layout

Field TIMEOUT

Timeout value in multiples of 65536 GPU clocks.

[PTM_RESOURCE_GROUP.PTM_WATCHDOG](#).TIMEOUT (T) is stored in bits [15:0] and is a 16-bit unsigned integer.

PTM_SLICE_POWER_STATE (0x0098)

Power state of slices.

Reports the power state of each slice assigned to this group; unassigned slices read zero.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	SLICE0	State of slice 0 power
1	SLICE1	State of slice 1 power
2	SLICE2	State of slice 2 power
3	SLICE3	State of slice 3 power
4	SLICE4	State of slice 4 power
5	SLICE5	State of slice 5 power
6	SLICE6	State of slice 6 power
7	SLICE7	State of slice 7 power
31:8	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE](#) layout

Field [SLICE0](#)

State of slice 0 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE0](#) (S0) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field [SLICE0](#) values

Field [SLICE1](#)

State of slice 1 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE1](#) (S1) is stored in bit [1] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field [SLICE1](#) values

Field [SLICE2](#)

State of slice 2 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE2](#) (S2) is stored in bit [2] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field [SLICE2](#) values

Field SLICE3

State of slice 3 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE3](#) (S3) is stored in bit [3] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field SLICE3 values

Field SLICE4

State of slice 4 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE4](#) (S4) is stored in bit [4] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field SLICE4 values

Field SLICE5

State of slice 5 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE5](#) (S5) is stored in bit [5] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field SLICE5 values

Field SLICE6

State of slice 6 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE6](#) (S6) is stored in bit [6] and is a 1-bit flag.

Value	Meaning
0	Slice power is off
1	Slice power is on

Field SLICE6 values

Field SLICE7

State of slice 7 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_STATE.SLICE7](#) (S7) is stored in bit [7] and is a 1-bit flag.

Value	Meaning
-------	---------

0	Slice power is off
1	Slice power is on

Field SLICE7 values

PTM_SLICE_POWER_SET (0x009C)

Set slice power mode.

Sets the power mode of a slice assigned to this group; fields for unassigned slices ignore writes and read zero. A write to a field is ignored if the slice is transitioning between power states indicated by a difference between the SET and STATE registers.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	SLICE0	Control slice 0 power	0
1	SLICE1	Control slice 1 power	0
2	SLICE2	Control slice 2 power	0
3	SLICE3	Control slice 3 power	0
4	SLICE4	Control slice 4 power	0
5	SLICE5	Control slice 5 power	0
6	SLICE6	Control slice 6 power	0
7	SLICE7	Control slice 7 power	0
31:8	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#) layout

Field SLICE0

Control slice 0 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET.SLICE0](#) (S0) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE0 values

Field SLICE1

Control slice 1 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET.SLICE1](#) (S1) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE1 values

Field SLICE2

Control slice 2 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE2 (S2) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE2 values

Field SLICE3

Control slice 3 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE3 (S3) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE3 values

Field SLICE4

Control slice 4 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE4 (S4) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE4 values

Field SLICE5

Control slice 5 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE5 (S5) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE5 values

Field SLICE6

Control slice 6 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE6 (S6) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE6 values

Field SLICE7

Control slice 7 power.

[PTM_RESOURCE_GROUP.PTM_SLICE_POWER_SET](#).SLICE7 (S7) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Turn slice power off
1	Turn slice power on

Field SLICE7 values

PTM_SLICE_CLOCK_STATE (0x00A0)

Clock state of slices.

Reports the clock state of each slice assigned to this group; unassigned slices read zero.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	SLICE0	State of slice 0 clock
1	SLICE1	State of slice 1 clock
2	SLICE2	State of slice 2 clock
3	SLICE3	State of slice 3 clock
4	SLICE4	State of slice 4 clock
5	SLICE5	State of slice 5 clock
6	SLICE6	State of slice 6 clock
7	SLICE7	State of slice 7 clock
31:8	Reserved (zero)	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#) layout

Field SLICE0

State of slice 0 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE0 (S0) is stored in bit [0] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE0 values

Field SLICE1

State of slice 1 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE1 (S1) is stored in bit [1] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE1 values

Field SLICE2

State of slice 2 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE2 (S2) is stored in bit [2] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE2 values

Field SLICE3

State of slice 3 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE3 (S3) is stored in bit [3] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE3 values

Field SLICE4

State of slice 4 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE4 (S4) is stored in bit [4] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE4 values

Field SLICE5

State of slice 5 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE5 (S5) is stored in bit [5] and is a 1-bit flag.

Value	Meaning
-------	---------

0	Slice clock is off
1	Slice clock is on

Field SLICE5 values

Field SLICE6

State of slice 6 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE6 (S6) is stored in bit [6] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE6 values

Field SLICE7

State of slice 7 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_STATE](#).SLICE7 (S7) is stored in bit [7] and is a 1-bit flag.

Value	Meaning
0	Slice clock is off
1	Slice clock is on

Field SLICE7 values

PTM_SLICE_CLOCK_SET (0x00A4)

Set slice clock mode.

Sets the clock mode for each slice assigned to this group; fields for unassigned slices ignore writes and read zero.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
1:0	SLICE0	Control slice 0 clock	Disable
3:2	SLICE1	Control slice 1 clock	Disable
5:4	SLICE2	Control slice 2 clock	Disable
7:6	SLICE3	Control slice 3 clock	Disable
9:8	SLICE4	Control slice 4 clock	Disable
11:10	SLICE5	Control slice 5 clock	Disable
13:12	SLICE6	Control slice 6 clock	Disable
15:14	SLICE7	Control slice 7 clock	Disable
31:16	Reserved (zero)	-	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET](#) layout

Field SLICE0

Control slice 0 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET.SLICE0](#) (S0) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE0 values

Field SLICE1

Control slice 1 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET.SLICE1](#) (S1) is stored in bits [3:2] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE1 values

Field SLICE2

Control slice 2 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET.SLICE2](#) (S2) is stored in bits [5:4] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE2 values

Field SLICE3

Control slice 3 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET.SLICE3](#) (S3) is stored in bits [7:6] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE3 values

Field SLICE4

Control slice 4 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET](#).SLICE4 (S4) is stored in bits [9:8] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE4 values

Field SLICE5

Control slice 5 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET](#).SLICE5 (S5) is stored in bits [11:10] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE5 values

Field SLICE6

Control slice 6 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET](#).SLICE6 (S6) is stored in bits [13:12] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE6 values

Field SLICE7

Control slice 7 clock.

[PTM_RESOURCE_GROUP.PTM_SLICE_CLOCK_SET](#).SLICE7 (S7) is stored in bits [15:14] and is a 2-bit enumeration of type PTM_CLOCK_MODE_ENUM. Its default value is Disable.

The field can contain the following values:

Value	Name	Meaning
0 (default)	Disable	Disable slice clock
1	Enable	Enable slice clock
2	Automatic	Control slice clock based on activity
3	Reserved	-

Field SLICE7 values

PTM_SLICE_RESET_STATE (0x00A8)

Reset state of slices.

Reports the reset state of each slice assigned to this group; unassigned slices read zero.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
0	SLICE0	Reset state of slice 0
1	SLICE1	Reset state of slice 1
2	SLICE2	Reset state of slice 2
3	SLICE3	Reset state of slice 3
4	SLICE4	Reset state of slice 4
5	SLICE5	Reset state of slice 5
6	SLICE6	Reset state of slice 6
7	SLICE7	Reset state of slice 7
31:8	Reserved	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE](#) layout

Field SLICE0

Reset state of slice 0.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE](#).SLICE0 (S0) is stored in bits [0] and is a 1-bit flag.

Value	Meaning
0	Slice 0 is not in reset
1	Slice 0 is in reset

Field SLICE0 values

Field SLICE1

Reset state of slice 1.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE.SLICE1](#) (S1) is stored in bits [1] and is a 1-bit flag.

Value	Meaning
0	Slice 1 is not in reset
1	Slice 1 is in reset

Field SLICE1 values

Field SLICE2

Reset state of slice 2.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE.SLICE2](#) (S2) is stored in bits [2] and is a 1-bit flag.

Value	Meaning
0	Slice 2 is not in reset
1	Slice 2 is in reset

Field SLICE2 values

Field SLICE3

Reset state of slice 3.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE.SLICE3](#) (S3) is stored in bits [3] and is a 1-bit flag.

Value	Meaning
0	Slice 3 is not in reset
1	Slice 3 is in reset

Field SLICE3 values

Field SLICE4

Reset state of slice 4.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE.SLICE4](#) (S4) is stored in bits [4] and is a 1-bit flag.

Value	Meaning
0	Slice 4 is not in reset
1	Slice 4 is in reset

Field SLICE4 values

Field SLICE5

Reset state of slice 5.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE.SLICE5](#) (S5) is stored in bits [5] and is a 1-bit flag.

Value	Meaning
-------	---------

0	Slice 5 is not in reset
1	Slice 5 is in reset

Field SLICE5 values

Field SLICE6

Reset state of slice 6.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE](#).SLICE6 (S6) is stored in bits [6] and is a 1-bit flag.

Value	Meaning
0	Slice 6 is not in reset
1	Slice 6 is in reset

Field SLICE6 values

Field SLICE7

Reset state of slice 7.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_STATE](#).SLICE7 (S7) is stored in bits [7] and is a 1-bit flag.

Value	Meaning
0	Slice 7 is not in reset
1	Slice 7 is in reset

Field SLICE7 values

PTM_SLICE_RESET_SET (0x00AC)

Set slice reset.

Sets the reset mode for slices assigned to this group; fields for unassigned slices ignore writes and read as zero. A write to a field is ignored if the slice is transitioning between reset states indicated by a difference between the SET and STATE registers.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	SLICE0	Reset slice 0	0
1	SLICE1	Reset slice 1	0
2	SLICE2	Reset slice 2	0
3	SLICE3	Reset slice 3	0
4	SLICE4	Reset slice 4	0
5	SLICE5	Reset slice 5	0
6	SLICE6	Reset slice 6	0
7	SLICE7	Reset slice 7	0
31:8	Reserved	-	-

Register [PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#) layout

Field SLICE0

Reset slice 0.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE0 (S0) is stored in bits [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 0
1	Reset slice 0

Field SLICE0 values

Field SLICE1

Reset slice 1.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE1 (S1) is stored in bits [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 1
1	Reset slice 1

Field SLICE1 values

Field SLICE2

Reset slice 2.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE2 (S2) is stored in bits [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 2
1	Reset slice 2

Field SLICE2 values

Field SLICE3

Reset slice 3.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE3 (S3) is stored in bits [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 3
1	Reset slice 3

Field SLICE3 values

Field SLICE4

Reset slice 4.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE4 (S4) is stored in bits [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Release slice 4
1	Reset slice 4

Field SLICE4 values

Field SLICE5

Reset slice 5.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE5 (S5) is stored in bits [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 5
1	Reset slice 5

Field SLICE5 values

Field SLICE6

Reset slice 6.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE6 (S6) is stored in bits [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 6
1	Reset slice 6

Field SLICE6 values

Field SLICE7

Reset slice 7.

[PTM_RESOURCE_GROUP.PTM_SLICE_RESET_SET](#).SLICE7 (S7) is stored in bits [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Release slice 7
1	Reset slice 7

Field SLICE7 values

PTM_AW0_MESSAGE (0x0100 - 0x011C)

Access window 0 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW1_MESSAGE (0x0120 - 0x013C)

Access window 1 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW2_MESSAGE (0x0140 - 0x015C)

Access window 2 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW3_MESSAGE (0x0160 - 0x017C)

Access window 3 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW4_MESSAGE (0x0180 - 0x019C)

Access window 4 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW5_MESSAGE (0x01A0 - 0x01BC)

Access window 5 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW6_MESSAGE (0x01C0 - 0x01DC)

Access window 6 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW7_MESSAGE (0x01E0 - 0x01FC)

Access window 7 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW8_MESSAGE (0x0200 - 0x021C)

Access window 8 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW9_MESSAGE (0x0220 - 0x023C)

Access window 9 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW10_MESSAGE (0x0240 - 0x025C)

Access window 10 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW11_MESSAGE (0x0260 - 0x027C)

Access window 11 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW12_MESSAGE (0x0280 - 0x029C)

Access window 12 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW13_MESSAGE (0x02A0 - 0x02BC)

Access window 13 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW14_MESSAGE (0x02C0 - 0x02DC)

Access window 14 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

PTM_AW15_MESSAGE (0x02E0 - 0x02FC)

Access window 15 message registers.

A set of registers used for communication between an instance of the driver and system software. Details on the [PTM_MESSAGE](#) sub-page.

Access to this Register is restricted to mode [rw](#).

C.1.9 Register sub-page PTM_SYSTEM

Partition manager system registers.

Address	Name	Usage	Access
0x0000	PTM_ID	Partition Manager ID	ro
0x0004	PTM_UNIT_FEATURES	Partition Manager features	ro
0x0008	PTM_SLICE_FEATURES	Slice features	ro
0x000C - 0x0010	PTM_SLICE_CORES	Cores per slice	ro
0x0014 - 0x003C	Reserved	-	-
0x0040 - 0x0048	PTM_IRQ_RAWSTAT	Status of system interrupts before masking	ro
0x004C - 0x0054	PTM_IRQ_CLEAR	Clear system interrupts	rw
0x0058 - 0x0060	PTM_UNCORRECTED_ERROR_IRQ_MASK	Enable and disable system interrupts indicating an uncorrected error	rw
0x0064 - 0x006C	PTM_DEFERRED_ERROR_IRQ_MASK	Enable and disable system interrupts indicating a deferred error	rw
0x0070 - 0x0078	PTM_UNCORRECTED_ERROR_IRQ_STATUS	Status of system interrupts indicating an uncorrected error after masking	ro
0x007C - 0x0084	PTM_DEFERRED_ERROR_IRQ_STATUS	Status of system interrupts indicating a deferred error after masking	ro

Address	Name	Usage	Access
0x0088 - 0x0090	PTM_IRQ_INJECTION	Inject system interrupts (before masking)	rw
0x0094	PTM_ERROR_FINGER_PRINT	Information indicating the cause of an error reported by an interrupt	rw
0x0098	PTM_GROUP_RESET_STATE	Reset state of each group	ro
0x009C	PTM_GROUP_RESET_SET	Reset for each group	rw
0x00A0	PTM_ERROR_RESPONSE	Response to errors	rw
0x00A4	PTM_PARITY_RESPONSE	Response to interface errors	rw
0x00A8 - 0x0FFC	Reserved	-	-
0x1000	PTM_AW0_STREAM_ID	Stream ID	rw
0x1004	PTM_AW0_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1008	PTM_AW1_STREAM_ID	Stream ID	rw
0x100C	PTM_AW1_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1010	PTM_AW2_STREAM_ID	Stream ID	rw
0x1014	PTM_AW2_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1018	PTM_AW3_STREAM_ID	Stream ID	rw
0x101C	PTM_AW3_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1020	PTM_AW4_STREAM_ID	Stream ID	rw
0x1024	PTM_AW4_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1028	PTM_AW5_STREAM_ID	Stream ID	rw
0x102C	PTM_AW5_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1030	PTM_AW6_STREAM_ID	Stream ID	rw
0x1034	PTM_AW6_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1038	PTM_AW7_STREAM_ID	Stream ID	rw
0x103C	PTM_AW7_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1040	PTM_AW8_STREAM_ID	Stream ID	rw
0x1044	PTM_AW8_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1048	PTM_AW9_STREAM_ID	Stream ID	rw
0x104C	PTM_AW9_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1050	PTM_AW10_STREAM_ID	Stream ID	rw
0x1054	PTM_AW10_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1058	PTM_AW11_STREAM_ID	Stream ID	rw
0x105C	PTM_AW11_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1060	PTM_AW12_STREAM_ID	Stream ID	rw
0x1064	PTM_AW12_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1068	PTM_AW13_STREAM_ID	Stream ID	rw
0x106C	PTM_AW13_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1070	PTM_AW14_STREAM_ID	Stream ID	rw
0x1074	PTM_AW14_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1078	PTM_AW15_STREAM_ID	Stream ID	rw

Address	Name	Usage	Access
0x107C	PTM_AW15_PROTECTED_STREAM_ID	Protected Stream ID	rw
0x1080 - 0xFFFD	Reserved	-	-

List of registers

PTM_ID (0x0000)

Partition Manager ID.

Identifies the version of the GPU. Each product may be distinguished by the ARCH_MAJOR and PRODUCT_MAJOR fields.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
3:0	VERSION_STATUS	Contains the status of the GPU release	2 (implementation defined)
11:4	VERSION_MINOR	Contains the minor release number of the GPU (the P part of an RnPn release number)	0 (implementation defined)
15:12	VERSION_MAJOR	Contains the major release number of the GPU (the R part of an RnPn release number)	0 (implementation defined)
19:16	PRODUCT_MAJOR	Contains a value indicating a product release within a product generation implementing a particular major version of the architecture	5 (implementation defined)
23:20	ARCH_REV	Contains the patch revision value for the version of the architecture implemented by this hardware	5 (implementation defined)
27:24	ARCH_MINOR	Contains the minor revision value for the version of the architecture implemented by this hardware	14 (implementation defined)
31:28	ARCH_MAJOR	Contains the major revision value for the version of the architecture implemented by this hardware	9 (implementation defined)

Register [PTM_SYSTEM.PTM_ID](#) layout

Field [VERSION_STATUS](#)

Contains the status of the GPU release.

[PTM_SYSTEM.PTM_ID.VERSION_STATUS](#) (VSTAT) is stored in bits [3:0] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	2

Configuration-specific field properties

This field contains the status of the GPU release. This has no defined values, but starts at 0 and increases by one for each release status (alpha, beta, EAC, etc.)

Field [VERSION_MINOR](#)

Contains the minor release number of the GPU (the P part of an RnPn release number).

[PTM_SYSTEM.PTM_ID.VERSION_MINOR](#) (VMIN) is stored in bits [11:4] and is a 8-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **VERSION_MAJOR**

Contains the major release number of the GPU (the R part of an RnPn release number).

[PTM_SYSTEM.PTM_ID.VERSION_MAJOR](#) (VMAJ) is stored in bits [15:12] and is a 4-bit unsigned integer. Its default value is 0 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	0

Configuration-specific field properties

Field **PRODUCT_MAJOR**

Contains a value indicating a product release within a product generation implementing a particular major version of the architecture.

[PTM_SYSTEM.PTM_ID.PRODUCT_MAJOR](#) (PMAJ) is stored in bits [19:16] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field **ARCH_REV**

Contains the patch revision value for the version of the architecture implemented by this hardware.

[PTM_SYSTEM.PTM_ID.ARCH_REV](#) (AREV) is stored in bits [23:20] and is a 4-bit unsigned integer. Its default value is 5 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	5

Configuration-specific field properties

Field **ARCH_MINOR**

Contains the minor revision value for the version of the architecture implemented by this hardware.

[PTM_SYSTEM.PTM_ID.ARCH_MINOR](#) (AMIN) is stored in bits [27:24] and is a 4-bit unsigned integer. Its default value is 14 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	14

Configuration-specific field properties

Field ARCH_MAJOR

Contains the major revision value for the version of the architecture implemented by this hardware.

[PTM_SYSTEM.PTM_ID.ARCH_MAJOR](#) (AMAJ) is stored in bits [31:28] and is a 4-bit unsigned integer. Its default value is 9 (implementation defined).

Configuration-dependent properties

Property	Configuration	Value
Default field value	default	9

Configuration-specific field properties

PTM_UNIT_FEATURES (0x0004)

Partition Manager features.

Identifies the partition manager features.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
3:0	PARTITIONS	Total number of partitions in the system	Implementation defined	1	4
7:4	Reserved (zero)	-	-	-	-
13:8	ACCESS_WINDOWS	Total number of access windows in the system	Implementation defined	1	16
31:14	Reserved (zero)	-	-	-	-

Register [PTM_SYSTEM.PTM_UNIT_FEATURES](#) layout

Field PARTITIONS

Total number of partitions in the system.

[PTM_SYSTEM.PTM_UNIT_FEATURES.PARTITIONS](#) (P) is stored in bits [3:0] and is a 4-bit unsigned integer. Its value must lie in the range from 1 to 4 inclusive. Its default value is Implementation defined.

Field ACCESS_WINDOWS

Total number of access windows in the system.

[PTM_SYSTEM.PTM_UNIT_FEATURES.ACCESS_WINDOWS](#) (W) is stored in bits [13:8] and is a 6-bit unsigned integer. Its value must lie in the range from 1 to 16 inclusive. Its default value is Implementation defined.

PTM_SLICE_FEATURES (0x0008)

Slice features.

Reports features of each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
7:0	CORE_MASK_STRIDE	Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice)	4
8	SLICE0_TILER	Type of tiler in slice 0	Implementation defined
9	SLICE1_TILER	Type of tiler in slice 1	Implementation defined
10	SLICE2_TILER	Type of tiler in slice 2	Implementation defined
11	SLICE3_TILER	Type of tiler in slice 3	Implementation defined
12	SLICE4_TILER	Type of tiler in slice 4	Implementation defined
13	SLICE5_TILER	Type of tiler in slice 5	Implementation defined
14	SLICE6_TILER	Type of tiler in slice 6	Implementation defined
15	SLICE7_TILER	Type of tiler in slice 7	Implementation defined
31:16	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_SLICE_FEATURES](#) layout

Field [CORE_MASK_STRIDE](#)

Stride between slices in GPU_CONTROL.SHADER_PRESENT (the number of bits allocated to each slice).

[PTM_SYSTEM.PTM_SLICE_FEATURES.CORE_MASK_STRIDE](#) (S) is stored in bits [7:0] and is a 8-bit unsigned integer. Its default value is 4.

Field [SLICE0_TILER](#)

Type of tiler in slice 0.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE0_TILER](#) (T0) is stored in bit [8] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field [SLICE0_TILER](#) values

Field [SLICE1_TILER](#)

Type of tiler in slice 1.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE1_TILER](#) (T1) is stored in bit [9] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler

1	High performance tiler
---	------------------------

Field SLICE1_TILER values

Field SLICE2_TILER

Type of tiler in slice 2.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE2_TILER](#) (T2) is stored in bit [10] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE2_TILER values

Field SLICE3_TILER

Type of tiler in slice 3.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE3_TILER](#) (T3) is stored in bit [11] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE3_TILER values

Field SLICE4_TILER

Type of tiler in slice 4.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE4_TILER](#) (T4) is stored in bit [12] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE4_TILER values

Field SLICE5_TILER

Type of tiler in slice 5.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE5_TILER](#) (T5) is stored in bit [13] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE5_TILER values

Field SLICE6_TILER

Type of tiler in slice 6.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE6_TILER](#) (T6) is stored in bit [14] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE6_TILER values

Field SLICE7_TILER

Type of tiler in slice 7.

[PTM_SYSTEM.PTM_SLICE_FEATURES.SLICE7_TILER](#) (T7) is stored in bit [15] and is a 1-bit flag. Its default value is Implementation defined.

Value	Meaning
0	Compact tiler
1	High performance tiler

Field SLICE7_TILER values

PTM_SLICE_CORES (0x000C - 0x0010)

Cores per slice.

Identifies the number of cores in each slice

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default	Min	Max
7:0	CORES0	Number of shader cores in slice 0	Implementation defined	0	3
15:8	CORES1	Number of shader cores in slice 1	Implementation defined	0	3
23:16	CORES2	Number of shader cores in slice 2	Implementation defined	0	3
31:24	CORES3	Number of shader cores in slice 3	Implementation defined	0	3
39:32	CORES4	Number of shader cores in slice 4	Implementation defined	0	3
47:40	CORES5	Number of shader cores in slice 5	Implementation defined	0	3
55:48	CORES6	Number of shader cores in slice 6	Implementation defined	0	3
63:56	CORES7	Number of shader cores in slice 7	Implementation defined	0	3

Register [PTM_SYSTEM.PTM_SLICE_CORES](#) layout

Field CORES0

Number of shader cores in slice 0.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES0](#) (C0) is stored in bits [7:0] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES1

Number of shader cores in slice 1.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES1](#) (C1) is stored in bits [15:8] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES2

Number of shader cores in slice 2.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES2](#) (C2) is stored in bits [23:16] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES3

Number of shader cores in slice 3.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES3](#) (C3) is stored in bits [31:24] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES4

Number of shader cores in slice 4.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES4](#) (C4) is stored in bits [39:32] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES5

Number of shader cores in slice 5.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES5](#) (C5) is stored in bits [47:40] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES6

Number of shader cores in slice 6.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES6](#) (C6) is stored in bits [55:48] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

Field CORES7

Number of shader cores in slice 7.

[PTM_SYSTEM.PTM_SLICE_CORES.CORES7](#) (C7) is stored in bits [63:56] and is a 8-bit unsigned integer. Its value must lie in the range from 0 to 3 inclusive. Its default value is Implementation defined.

PTM_IRQ_RAWSTAT (0x0040 - 0x0048)

Status of system interrupts before masking.

Report of system interrupts before masking. This register is reset by PTM_GLOBAL_RESET but not by PTM_RECOVERY_RESET.

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Error on AXI-A	0
1	AXI_B_PARITY	Error on AXI-B	0
2	AXI_C_PARITY	Error on AXI-C	0
3	GENERAL_HARDWARE	General hardware error	0
4	SLICE0_PARITY	Parity error on slice 0 memory access	0
5	SLICE1_PARITY	Parity error on slice 1 memory access	0
6	SLICE2_PARITY	Parity error on slice 2 memory access	0
7	SLICE3_PARITY	Parity error on slice 3 memory access	0
8	SLICE4_PARITY	Parity error on slice 4 memory access	0
9	SLICE5_PARITY	Parity error on slice 5 memory access	0
10	SLICE6_PARITY	Parity error on slice 6 memory access	0
11	SLICE7_PARITY	Parity error on slice 7 memory access	0
12	SLICE0_ISOLATION	Isolation error on slice 0	0
13	SLICE1_ISOLATION	Isolation error on slice 1	0
14	SLICE2_ISOLATION	Isolation error on slice 2	0
15	SLICE3_ISOLATION	Isolation error on slice 3	0
16	SLICE4_ISOLATION	Isolation error on slice 4	0
17	SLICE5_ISOLATION	Isolation error on slice 5	0
18	SLICE6_ISOLATION	Isolation error on slice 6	0
19	SLICE7_ISOLATION	Isolation error on slice 7	0
27:20	SLICE_BIST	Slice BIST error	0
28	INVALID_BUS	Transaction received for address range that is not valid for that bus	0
31:21	Reserved (zero)		
32	GROUP0_WATCHDOG	Watchdog error for group 0	0
33	GROUP1_WATCHDOG	Watchdog error for group 1	0
34	GROUP2_WATCHDOG	Watchdog error for group 2	0
35	GROUP3_WATCHDOG	Watchdog error for group 3	0
36	PARTITION0_LOCKED_ACCESS	Attempt to access a locked partition register	0
37	PARTITION1_LOCKED_ACCESS	Attempt to access a locked partition register	0
38	PARTITION2_LOCKED_ACCESS	Attempt to access a locked partition register	0
39	PARTITION3_LOCKED_ACCESS	Attempt to access a locked partition register	0
40	PARTITION0_LOCKED_BIST	Attempt to access a locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Attempt to access a locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Attempt to access a locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Attempt to access a locked BIST controller	0
44	PARTITION0_MTCRC	Memory transaction CRC error	0
45	PARTITION1_MTCRC	Memory transaction CRC error	0
46	PARTITION2_MTCRC	Memory transaction CRC error	0
47	PARTITION3_MTCRC	Memory transaction CRC error	0
63:48	Reserved (zero)	-	-

Bits	Name	Usage	Default
64	AW0_INVALID_ACCESS	Attempt to access disabled window	0
65	AW1_INVALID_ACCESS	Attempt to access disabled window	0
66	AW2_INVALID_ACCESS	Attempt to access disabled window	0
67	AW3_INVALID_ACCESS	Attempt to access disabled window	0
68	AW4_INVALID_ACCESS	Attempt to access disabled window	0
69	AW5_INVALID_ACCESS	Attempt to access disabled window	0
70	AW6_INVALID_ACCESS	Attempt to access disabled window	0
71	AW7_INVALID_ACCESS	Attempt to access disabled window	0
72	AW8_INVALID_ACCESS	Attempt to access disabled window	0
73	AW9_INVALID_ACCESS	Attempt to access disabled window	0
74	AW10_INVALID_ACCESS	Attempt to access disabled window	0
75	AW11_INVALID_ACCESS	Attempt to access disabled window	0
76	AW12_INVALID_ACCESS	Attempt to access disabled window	0
77	AW13_INVALID_ACCESS	Attempt to access disabled window	0
78	AW14_INVALID_ACCESS	Attempt to access disabled window	0
79	AW15_INVALID_ACCESS	Attempt to access disabled window	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_IRQ_RAWSTAT](#) layout

Field **AXI_A_PARITY**

Error on AXI-A.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.AXI_A_PARITY](#) (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field **AXI_A_PARITY** values

Field **AXI_B_PARITY**

Error on AXI-B.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.AXI_B_PARITY](#) (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field **AXI_B_PARITY** values

Field **AXI_C_PARITY**

Error on AXI-C.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.AXI_C_PARITY](#) (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_C_PARITY values

Field GENERAL_HARDWARE

General hardware error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Parity error on slice 0 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_PARITY values

Field SLICE1_PARITY

Parity error on slice 1 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_PARITY values

Field SLICE2_PARITY

Parity error on slice 2 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE2_PARITY (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_PARITY values

Field SLICE3_PARITY

Parity error on slice 3 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE3_PARITY (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_PARITY values

Field SLICE4_PARITY

Parity error on slice 4 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE4_PARITY (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_PARITY values

Field SLICE5_PARITY

Parity error on slice 5 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE5_PARITY (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_PARITY values

Field SLICE6_PARITY

Parity error on slice 6 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE6_PARITY (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_PARITY values

Field SLICE7_PARITY

Parity error on slice 7 memory access.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE7_PARITY (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Isolation error on slice 0.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE0_ISOLATION (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Isolation error on slice 1.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Isolation error on slice 2.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Isolation error on slice 3.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Isolation error on slice 4.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Isolation error on slice 5.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Isolation error on slice 6.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Isolation error on slice 7.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_ISOLATION values

Field SLICE_BIST

Slice BIST error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Transaction received for address range that is not valid for that bus.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.INVALID_BUS](#) (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Watchdog error for group 0.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.GROUP0_WATCHDOG](#) (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Watchdog error for group 1.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.GROUP1_WATCHDOG](#) (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Watchdog error for group 2.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.GROUP2_WATCHDOG](#) (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Watchdog error for group 3.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT.GROUP3_WATCHDOG](#) (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_MTCRC values

Field AW0_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW0_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW0_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).AW15_INVALID_ACCESS (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW15_INVALID_ACCESS values

PTM_IRQ_CLEAR (0x004C - 0x0054)

Clear system interrupts.

Write '1' to clear the corresponding interrupt; this register always reads as zero

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Clear error on AXI-A	0
1	AXI_B_PARITY	Clear error on AXI-B	0
2	AXI_C_PARITY	Clear error on AXI-C	0
3	GENERAL_HARDWARE	Clear general hardware error signal	0
4	SLICE0_PARITY	Clear parity error on slice 0	0
5	SLICE1_PARITY	Clear parity error on slice 1	0
6	SLICE2_PARITY	Clear parity error on slice 2	0
7	SLICE3_PARITY	Clear parity error on slice 3	0
8	SLICE4_PARITY	Clear parity error on slice 4	0
9	SLICE5_PARITY	Clear parity error on slice 5	0
10	SLICE6_PARITY	Clear parity error on slice 6	0
11	SLICE7_PARITY	Clear parity error on slice 7	0
12	SLICE0_ISOLATION	Clear isolation error on slice 0	0
13	SLICE1_ISOLATION	Clear isolation error on slice 1	0
14	SLICE2_ISOLATION	Clear isolation error on slice 2	0
15	SLICE3_ISOLATION	Clear isolation error on slice 3	0
16	SLICE4_ISOLATION	Clear isolation error on slice 4	0
17	SLICE5_ISOLATION	Clear isolation error on slice 5	0
18	SLICE6_ISOLATION	Clear isolation error on slice 6	0
19	SLICE7_ISOLATION	Clear isolation error on slice 7	0
27:20	SLICE_BIST	Clear slice BIST error	0
28	INVALID_BUS	Clear invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Clear group 0 watchdog error	0
33	GROUP1_WATCHDOG	Clear group 1 watchdog error	0

Bits	Name	Usage	Default
34	GROUP2_WATCHDOG	Clear group 2 watchdog error	0
35	GROUP3_WATCHDOG	Clear group 3 watchdog error	0
36	PARTITION0_LOCKED_ACCESS	Clear locked partition error	0
37	PARTITION1_LOCKED_ACCESS	Clear locked partition error	0
38	PARTITION2_LOCKED_ACCESS	Clear locked partition error	0
39	PARTITION3_LOCKED_ACCESS	Clear locked partition error	0
40	PARTITION0_LOCKED_BIST	Clear locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Clear locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Clear locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Clear locked BIST controller	0
44	PARTITION0_MTCRC	Clear memory transaction CRC error	0
45	PARTITION1_MTCRC	Clear memory transaction CRC error	0
46	PARTITION2_MTCRC	Clear memory transaction CRC error	0
47	PARTITION3_MTCRC	Clear memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Clear disabled access window error	0
65	AW1_INVALID_ACCESS	Clear disabled access window error	0
66	AW2_INVALID_ACCESS	Clear disabled access window error	0
67	AW3_INVALID_ACCESS	Clear disabled access window error	0
68	AW4_INVALID_ACCESS	Clear disabled access window error	0
69	AW5_INVALID_ACCESS	Clear disabled access window error	0
70	AW6_INVALID_ACCESS	Clear disabled access window error	0
71	AW7_INVALID_ACCESS	Clear disabled access window error	0
72	AW8_INVALID_ACCESS	Clear disabled access window error	0
73	AW9_INVALID_ACCESS	Clear disabled access window error	0
74	AW10_INVALID_ACCESS	Clear disabled access window error	0
75	AW11_INVALID_ACCESS	Clear disabled access window error	0
76	AW12_INVALID_ACCESS	Clear disabled access window error	0
77	AW13_INVALID_ACCESS	Clear disabled access window error	0
78	AW14_INVALID_ACCESS	Clear disabled access window error	0
79	AW15_INVALID_ACCESS	Clear disabled access window error	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_IRQ_CLEAR](#) layout

Field AXI_A_PARITY

Clear error on AXI-A.

[PTM_SYSTEM.PTM_IRQ_CLEAR.AXI_A_PARITY](#) (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing

1	Clear error
---	-------------

Field AXI_A_PARITY values

Field AXI_B_PARITY

Clear error on AXI-B.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AXI_B_PARITY (BO) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AXI_B_PARITY values

Field AXI_C_PARITY

Clear error on AXI-C.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AXI_C_PARITY (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AXI_C_PARITY values

Field GENERAL_HARDWARE

Clear general hardware error signal.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Clear parity error on slice 0.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE0_PARITY values

Field SLICE1_PARITY

Clear parity error on slice 1.

[PTM_SYSTEM.PTM_IRQ_CLEAR.SLICE1_PARITY](#) (SP1) is stored in bit [5] and is a 1-bit flag.
Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE1_PARITY values

Field SLICE2_PARITY

Clear parity error on slice 2.

[PTM_SYSTEM.PTM_IRQ_CLEAR.SLICE2_PARITY](#) (SP2) is stored in bit [6] and is a 1-bit flag.
Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE2_PARITY values

Field SLICE3_PARITY

Clear parity error on slice 3.

[PTM_SYSTEM.PTM_IRQ_CLEAR.SLICE3_PARITY](#) (SP3) is stored in bit [7] and is a 1-bit flag.
Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE3_PARITY values

Field SLICE4_PARITY

Clear parity error on slice 4.

[PTM_SYSTEM.PTM_IRQ_CLEAR.SLICE4_PARITY](#) (SP4) is stored in bit [8] and is a 1-bit flag.
Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE4_PARITY values

Field SLICE5_PARITY

Clear parity error on slice 5.

[PTM_SYSTEM.PTM_IRQ_CLEAR.SLICE5_PARITY](#) (SP5) is stored in bit [9] and is a 1-bit flag.
Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE5_PARITY values

Field SLICE6_PARITY

Clear parity error on slice 6.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE6_PARITY (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE6_PARITY values

Field SLICE7_PARITY

Clear parity error on slice 7.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE7_PARITY (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Clear isolation error on slice 0.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE0_ISOLATION (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Clear isolation error on slice 1.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Clear isolation error on slice 2.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Clear isolation error on slice 3.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Clear isolation error on slice 4.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Clear isolation error on slice 5.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Clear isolation error on slice 6.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Clear isolation error on slice 7.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field SLICE7_ISOLATION values

Field SLICE_BIST

Clear slice BIST error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Clear invalid bus error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Clear group 0 watchdog error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Clear group 1 watchdog error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Do nothing
1	Clear error

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Clear group 2 watchdog error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Clear group 3 watchdog error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Clear locked partition error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Clear locked partition error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Clear locked partition error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Clear locked partition error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Clear locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Clear locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Clear locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Clear locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Clear memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Clear memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Clear memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Clear memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field PARTITION3_MTCRC values

Field AWO_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AWO_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AWO_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Do nothing
1	Clear error

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Clear disabled access window error.

[PTM_SYSTEM.PTM_IRQ_CLEAR](#).AW15_INVALID_ACCESS (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Do nothing
1	Clear error

Field AW15_INVALID_ACCESS values

PTM_UNCORRECTED_ERROR_IRQ_MASK (0x0058 - 0x0060)

Enable and disable system interrupts indicating an uncorrected error.

Set to '1' to enable the corresponding interrupt

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Enable error reporting for AXI-A	0
1	AXI_B_PARITY	Enable error reporting for AXI-B	0
2	AXI_C	Enable error reporting for AXI-C	0
3	GENERAL_HARDWARE	Enable general hardware error reporting	0
4	SLICE0_PARITY	Enable parity error reporting for slice 0	0
5	SLICE1_PARITY	Enable parity error reporting for slice 1	0
6	SLICE2_PARITY	Enable parity error reporting for slice 2	0
7	SLICE3_PARITY	Enable parity error reporting for slice 3	0
8	SLICE4_PARITY	Enable parity error reporting for slice 4	0

Bits	Name	Usage	Default
9	SLICE5_PARITY	Enable parity error reporting for slice 5	0
10	SLICE6_PARITY	Enable parity error reporting for slice 6	0
11	SLICE7_PARITY	Enable parity error reporting for slice 7	0
12	SLICE0_ISOLATION	Enable isolation error reporting for slice 0	0
13	SLICE1_ISOLATION	Enable isolation error reporting for slice 1	0
14	SLICE2_ISOLATION	Enable isolation error reporting for slice 2	0
15	SLICE3_ISOLATION	Enable isolation error reporting for slice 3	0
16	SLICE4_ISOLATION	Enable isolation error reporting for slice 4	0
17	SLICE5_ISOLATION	Enable isolation error reporting for slice 5	0
18	SLICE6_ISOLATION	Enable isolation error reporting for slice 6	0
19	SLICE7_ISOLATION	Enable isolation error reporting for slice 7	0
27:20	SLICE_BIST	Enable slice BIST error reporting	0
28	INVALID_BUS	Enable invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Enable group 0 watchdog error	0
33	GROUP1_WATCHDOG	Enable group 1 watchdog error	0
34	GROUP2_WATCHDOG	Enable group 2 watchdog error	0
35	GROUP3_WATCHDOG	Enable group 3 watchdog error	0
36	PARTITION0_LOCKED_ACCESS	Enable locked partition error	0
37	PARTITION1_LOCKED_ACCESS	Enable locked partition error	0
38	PARTITION2_LOCKED_ACCESS	Enable locked partition error	0
39	PARTITION3_LOCKED_ACCESS	Enable locked partition error	0
40	PARTITION0_LOCKED_BIST	Enable locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Enable locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Enable locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Enable locked BIST controller	0
44	PARTITION0_MTCRC	Enable memory transaction CRC error	0
45	PARTITION1_MTCRC	Enable memory transaction CRC error	0
46	PARTITION2_MTCRC	Enable memory transaction CRC error	0
47	PARTITION3_MTCRC	Enable memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Enable disabled access window error	0
65	AW1_INVALID_ACCESS	Enable disabled access window error	0
66	AW2_INVALID_ACCESS	Enable disabled access window error	0
67	AW3_INVALID_ACCESS	Enable disabled access window error	0
68	AW4_INVALID_ACCESS	Enable disabled access window error	0
69	AW5_INVALID_ACCESS	Enable disabled access window error	0
70	AW6_INVALID_ACCESS	Enable disabled access window error	0
71	AW7_INVALID_ACCESS	Enable disabled access window error	0
72	AW8_INVALID_ACCESS	Enable disabled access window error	0

Bits	Name	Usage	Default
73	AW9_INVALID_ACCESS	Enable disabled access window error	0
74	AW10_INVALID_ACCESS	Enable disabled access window error	0
75	AW11_INVALID_ACCESS	Enable disabled access window error	0
76	AW12_INVALID_ACCESS	Enable disabled access window error	0
77	AW13_INVALID_ACCESS	Enable disabled access window error	0
78	AW14_INVALID_ACCESS	Enable disabled access window error	0
79	AW15_INVALID_ACCESS	Enable disabled access window error	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#) layout

Field **AXI_A_PARITY**

Enable error reporting for AXI-A.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.AXI_A_PARITY](#) (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field **AXI_A_PARITY** values

Field **AXI_B_PARITY**

Enable error reporting for AXI-B.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.AXI_B_PARITY](#) (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field **AXI_B_PARITY** values

Field **AXI_C**

Enable error reporting for AXI-C.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.AXI_C](#) (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field **AXI_C** values

Field **GENERAL_HARDWARE**

Enable general hardware error reporting.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.GENERAL_HARDWARE](#) (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Enable parity error reporting for slice 0.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE0_PARITY values

Field SLICE1_PARITY

Enable parity error reporting for slice 1.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE1_PARITY values

Field SLICE2_PARITY

Enable parity error reporting for slice 2.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE2_PARITY (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE2_PARITY values

Field SLICE3_PARITY

Enable parity error reporting for slice 3.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE3_PARITY (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE3_PARITY values

Field SLICE4_PARITY

Enable parity error reporting for slice 4.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.SLICE4_PARITY](#) (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE4_PARITY values

Field SLICE5_PARITY

Enable parity error reporting for slice 5.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.SLICE5_PARITY](#) (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE5_PARITY values

Field SLICE6_PARITY

Enable parity error reporting for slice 6.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.SLICE6_PARITY](#) (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE6_PARITY values

Field SLICE7_PARITY

Enable parity error reporting for slice 7.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.SLICE7_PARITY](#) (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Enable isolation error reporting for slice 0.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK.SLICE0_ISOLATION](#) (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Disable
1	Enable

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Enable isolation error reporting for slice 1.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Enable isolation error reporting for slice 2.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Enable isolation error reporting for slice 3.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Enable isolation error reporting for slice 4.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Enable isolation error reporting for slice 5.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Enable isolation error reporting for slice 6.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Enable isolation error reporting for slice 7.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE7_ISOLATION values

Field SLICE_BIST

Enable slice BIST error reporting.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Enable invalid bus error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Enable group 0 watchdog error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Enable group 1 watchdog error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Enable group 2 watchdog error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Enable group 3 watchdog error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION0_LOCKED_BIST (LBO) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Disable
1	Enable

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_MTCRC values

Field AWO_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AWO_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AWO_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_MASK](#).AW15_INVALID_ACCESS (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW15_INVALID_ACCESS values

PTM_DEFERRED_ERROR_IRQ_MASK (0x0064 - 0x006C)

Enable and disable system interrupts indicating a deferred error.

Set to '1' to enable the corresponding interrupt

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Enable error reporting for AXI-A	0
1	AXI_B_PARITY	Enable error reporting for AXI-B	0
2	AXI_C	Enable error reporting for AXI-C	0
3	GENERAL_HARDWARE	Enable general hardware error reporting	0
4	SLICE0_PARITY	Enable parity error reporting for slice 0	0
5	SLICE1_PARITY	Enable parity error reporting for slice 1	0
6	SLICE2_PARITY	Enable parity error reporting for slice 2	0
7	SLICE3_PARITY	Enable parity error reporting for slice 3	0
8	SLICE4_PARITY	Enable parity error reporting for slice 4	0
9	SLICE5_PARITY	Enable parity error reporting for slice 5	0
10	SLICE6_PARITY	Enable parity error reporting for slice 6	0
11	SLICE7_PARITY	Enable parity error reporting for slice 7	0
12	SLICE0_ISOLATION	Enable isolation error reporting for slice 0	0
13	SLICE1_ISOLATION	Enable isolation error reporting for slice 1	0
14	SLICE2_ISOLATION	Enable isolation error reporting for slice 2	0
15	SLICE3_ISOLATION	Enable isolation error reporting for slice 3	0
16	SLICE4_ISOLATION	Enable isolation error reporting for slice 4	0
17	SLICE5_ISOLATION	Enable isolation error reporting for slice 5	0
18	SLICE6_ISOLATION	Enable isolation error reporting for slice 6	0
19	SLICE7_ISOLATION	Enable isolation error reporting for slice 7	0
27:20	SLICE_BIST	Enable slice BIST error reporting	0
28	INVALID_BUS	Enable invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Enable group 0 watchdog error	0
33	GROUP1_WATCHDOG	Enable group 1 watchdog error	0
34	GROUP2_WATCHDOG	Enable group 2 watchdog error	0
35	GROUP3_WATCHDOG	Enable group 3 watchdog error	0
36	PARTITION0_LOCKED_ACCESS	Enable locked partition error	0
37	PARTITION1_LOCKED_ACCESS	Enable locked partition error	0
38	PARTITION2_LOCKED_ACCESS	Enable locked partition error	0
39	PARTITION3_LOCKED_ACCESS	Enable locked partition error	0
40	PARTITION0_LOCKED_BIST	Enable locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Enable locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Enable locked BIST controller	0

Bits	Name	Usage	Default
43	PARTITION3_LOCKED_BIST	Enable locked BIST controller	0
44	PARTITION0_MTCRC	Enable memory transaction CRC error	0
45	PARTITION1_MTCRC	Enable memory transaction CRC error	0
46	PARTITION2_MTCRC	Enable memory transaction CRC error	0
47	PARTITION3_MTCRC	Enable memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Enable disabled access window error	0
65	AW1_INVALID_ACCESS	Enable disabled access window error	0
66	AW2_INVALID_ACCESS	Enable disabled access window error	0
67	AW3_INVALID_ACCESS	Enable disabled access window error	0
68	AW4_INVALID_ACCESS	Enable disabled access window error	0
69	AW5_INVALID_ACCESS	Enable disabled access window error	0
70	AW6_INVALID_ACCESS	Enable disabled access window error	0
71	AW7_INVALID_ACCESS	Enable disabled access window error	0
72	AW8_INVALID_ACCESS	Enable disabled access window error	0
73	AW9_INVALID_ACCESS	Enable disabled access window error	0
74	AW10_INVALID_ACCESS	Enable disabled access window error	0
75	AW11_INVALID_ACCESS	Enable disabled access window error	0
76	AW12_INVALID_ACCESS	Enable disabled access window error	0
77	AW13_INVALID_ACCESS	Enable disabled access window error	0
78	AW14_INVALID_ACCESS	Enable disabled access window error	0
79	AW15_INVALID_ACCESS	Enable disabled access window error	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#) layout

Field AXI_A_PARITY

Enable error reporting for AXI-A.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.AXI_A_PARITY](#) (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AXI_A_PARITY values

Field AXI_B_PARITY

Enable error reporting for AXI-B.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.AXI_B_PARITY](#) (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable

1	Enable
---	--------

Field AXI_B_PARITY values

Field AXI_C

Enable error reporting for AXI-C.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AXI_C (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AXI_C values

Field GENERAL_HARDWARE

Enable general hardware error reporting.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Enable parity error reporting for slice 0.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE0_PARITY values

Field SLICE1_PARITY

Enable parity error reporting for slice 1.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE1_PARITY values

Field SLICE2_PARITY

Enable parity error reporting for slice 2.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE2_PARITY](#) (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE2_PARITY values

Field SLICE3_PARITY

Enable parity error reporting for slice 3.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE3_PARITY](#) (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE3_PARITY values

Field SLICE4_PARITY

Enable parity error reporting for slice 4.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE4_PARITY](#) (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE4_PARITY values

Field SLICE5_PARITY

Enable parity error reporting for slice 5.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE5_PARITY](#) (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE5_PARITY values

Field SLICE6_PARITY

Enable parity error reporting for slice 6.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE6_PARITY](#) (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE6_PARITY values

Field SLICE7_PARITY

Enable parity error reporting for slice 7.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE7_PARITY](#) (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Enable isolation error reporting for slice 0.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE0_ISOLATION](#) (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Enable isolation error reporting for slice 1.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE1_ISOLATION](#) (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Enable isolation error reporting for slice 2.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE2_ISOLATION](#) (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Enable isolation error reporting for slice 3.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.SLICE3_ISOLATION](#) (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Enable isolation error reporting for slice 4.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Enable isolation error reporting for slice 5.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Enable isolation error reporting for slice 6.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Enable isolation error reporting for slice 7.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field SLICE7_ISOLATION values

Field SLICE_BIST

Enable slice BIST error reporting.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Enable invalid bus error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Enable group 0 watchdog error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Enable group 1 watchdog error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Enable group 2 watchdog error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Disable
1	Enable

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Enable group 3 watchdog error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Enable locked partition error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Enable locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Enable memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field PARTITION3_MTCRC values

Field AW0_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW0_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW0_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	Disable
1	Enable

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.AW14_INVALID_ACCESS](#) (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Enable disabled access window error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_MASK.AW15_INVALID_ACCESS](#) (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Disable
1	Enable

Field AW15_INVALID_ACCESS values

PTM_UNCORRECTED_ERROR_IRQ_STATUS (0x0070 - 0x0078)

Status of system interrupts indicating an uncorrected error after masking.

Report of system interrupts after masking; an interrupt is raised if any bit is '1'

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Error on bus AXI-A	0
1	AXI_B_PARITY	Error on bus AXI-B	0
2	AXI_C_PARITY	Error on bus AXI-C	0
3	GENERAL_HARDWARE	General hardware error	0
4	SLICE0_PARITY	Parity error on bus slice 0	0
5	SLICE1_PARITY	Parity error on bus slice 1	0
6	SLICE2_PARITY	Parity error on bus slice 2	0
7	SLICE3_PARITY	Parity error on bus slice 3	0
8	SLICE4_PARITY	Parity error on bus slice 4	0
9	SLICE5_PARITY	Parity error on bus slice 5	0
10	SLICE6_PARITY	Parity error on bus slice 6	0
11	SLICE7_PARITY	Parity error on bus slice 7	0
12	SLICE0_ISOLATION	Isolation error on bus slice 0	0
13	SLICE1_ISOLATION	Isolation error on bus slice 1	0
14	SLICE2_ISOLATION	Isolation error on bus slice 2	0
15	SLICE3_ISOLATION	Isolation error on bus slice 3	0
16	SLICE4_ISOLATION	Isolation error on bus slice 4	0
17	SLICE5_ISOLATION	Isolation error on bus slice 5	0

Bits	Name	Usage	Default
18	SLICE6_ISOLATION	Isolation error on bus slice 6	0
19	SLICE7_ISOLATION	Isolation error on bus slice 7	0
27:20	SLICE_BIST	Slice BIST error	0
28	INVALID_BUS	Enable invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Watchdog error for group 0	0
33	GROUP1_WATCHDOG	Watchdog error for group 1	0
34	GROUP2_WATCHDOG	Watchdog error for group 2	0
35	GROUP3_WATCHDOG	Watchdog error for group 3	0
36	PARTITION0_LOCKED_ACCESS	Attempt to access a locked partition register	0
37	PARTITION1_LOCKED_ACCESS	Attempt to access a locked partition register	0
38	PARTITION2_LOCKED_ACCESS	Attempt to access a locked partition register	0
39	PARTITION3_LOCKED_ACCESS	Attempt to access a locked partition register	0
40	PARTITION0_LOCKED_BIST	Attempt to access a locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Attempt to access a locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Attempt to access a locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Attempt to access a locked BIST controller	0
44	PARTITION0_MTCRC	Memory transaction CRC error	0
45	PARTITION1_MTCRC	Memory transaction CRC error	0
46	PARTITION2_MTCRC	Memory transaction CRC error	0
47	PARTITION3_MTCRC	Memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Attempt to access disabled window	0
65	AW1_INVALID_ACCESS	Attempt to access disabled window	0
66	AW2_INVALID_ACCESS	Attempt to access disabled window	0
67	AW3_INVALID_ACCESS	Attempt to access disabled window	0
68	AW4_INVALID_ACCESS	Attempt to access disabled window	0
69	AW5_INVALID_ACCESS	Attempt to access disabled window	0
70	AW6_INVALID_ACCESS	Attempt to access disabled window	0
71	AW7_INVALID_ACCESS	Attempt to access disabled window	0
72	AW8_INVALID_ACCESS	Attempt to access disabled window	0
73	AW9_INVALID_ACCESS	Attempt to access disabled window	0
74	AW10_INVALID_ACCESS	Attempt to access disabled window	0
75	AW11_INVALID_ACCESS	Attempt to access disabled window	0
76	AW12_INVALID_ACCESS	Attempt to access disabled window	0
77	AW13_INVALID_ACCESS	Attempt to access disabled window	0
78	AW14_INVALID_ACCESS	Attempt to access disabled window	0
79	AW15_INVALID_ACCESS	Attempt to access disabled window	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#) layout

Field AXI_A_PARITY

Error on bus AXI-A.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AXI_A_PARITY (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_A_PARITY values

Field AXI_B_PARITY

Error on bus AXI-B.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AXI_B_PARITY (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_B_PARITY values

Field AXI_C_PARITY

Error on bus AXI-C.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AXI_C_PARITY (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_C_PARITY values

Field GENERAL_HARDWARE

General hardware error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Parity error on bus slice 0.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field SLICE0_PARITY values

Field SLICE1_PARITY

Parity error on bus slice 1.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_PARITY values

Field SLICE2_PARITY

Parity error on bus slice 2.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE2_PARITY (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_PARITY values

Field SLICE3_PARITY

Parity error on bus slice 3.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE3_PARITY (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_PARITY values

Field SLICE4_PARITY

Parity error on bus slice 4.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE4_PARITY (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_PARITY values

Field SLICE5_PARITY

Parity error on bus slice 5.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE5_PARITY (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_PARITY values

Field SLICE6_PARITY

Parity error on bus slice 6.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE6_PARITY (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_PARITY values

Field SLICE7_PARITY

Parity error on bus slice 7.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE7_PARITY (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Isolation error on bus slice 0.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE0_ISOLATION (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Isolation error on bus slice 1.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Isolation error on bus slice 2.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Isolation error on bus slice 3.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Isolation error on bus slice 4.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Isolation error on bus slice 5.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Isolation error on bus slice 6.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Isolation error on bus slice 7.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_ISOLATION values

Field SLICE_BIST

Slice BIST error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Enable invalid bus error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Watchdog error for group 0.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Watchdog error for group 1.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Watchdog error for group 2.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Watchdog error for group 3.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_MTCRC values

Field AW0_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW0_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW0_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_UNCORRECTED_ERROR_IRQ_STATUS](#).AW15_INVALID_ACCESS (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW15_INVALID_ACCESS values

PTM_DEFERRED_ERROR_IRQ_STATUS (0x007C - 0x0084)

Status of system interrupts indicating a deferred error after masking.

Report of system interrupts after masking; an interrupt is raised if any bit is '1'

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Error on bus AXI-A	0
1	AXI_B_PARITY	Error on bus AXI-B	0
2	AXI_C_PARITY	Error on bus AXI-C	0

Bits	Name	Usage	Default
3	GENERAL_HARDWARE	General hardware error	0
4	SLICE0_PARITY	Parity error on bus slice 0	0
5	SLICE1_PARITY	Parity error on bus slice 1	0
6	SLICE2_PARITY	Parity error on bus slice 2	0
7	SLICE3_PARITY	Parity error on bus slice 3	0
8	SLICE4_PARITY	Parity error on bus slice 4	0
9	SLICE5_PARITY	Parity error on bus slice 5	0
10	SLICE6_PARITY	Parity error on bus slice 6	0
11	SLICE7_PARITY	Parity error on bus slice 7	0
12	SLICE0_ISOLATION	Isolation error on bus slice 0	0
13	SLICE1_ISOLATION	Isolation error on bus slice 1	0
14	SLICE2_ISOLATION	Isolation error on bus slice 2	0
15	SLICE3_ISOLATION	Isolation error on bus slice 3	0
16	SLICE4_ISOLATION	Isolation error on bus slice 4	0
17	SLICE5_ISOLATION	Isolation error on bus slice 5	0
18	SLICE6_ISOLATION	Isolation error on bus slice 6	0
19	SLICE7_ISOLATION	Isolation error on bus slice 7	0
27:20	SLICE_BIST	Slice BIST error	0
28	INVALID_BUS	Enable invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Watchdog error for group 0	0
33	GROUP1_WATCHDOG	Watchdog error for group 1	0
34	GROUP2_WATCHDOG	Watchdog error for group 2	0
35	GROUP3_WATCHDOG	Watchdog error for group 3	0
36	PARTITION0_LOCKED_ACCESS	Attempt to access a locked partition register	0
37	PARTITION1_LOCKED_ACCESS	Attempt to access a locked partition register	0
38	PARTITION2_LOCKED_ACCESS	Attempt to access a locked partition register	0
39	PARTITION3_LOCKED_ACCESS	Attempt to access a locked partition register	0
40	PARTITION0_LOCKED_BIST	Attempt to access a locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Attempt to access a locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Attempt to access a locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Attempt to access a locked BIST controller	0
44	PARTITION0_MTCRC	Memory transaction CRC error	0
45	PARTITION1_MTCRC	Memory transaction CRC error	0
46	PARTITION2_MTCRC	Memory transaction CRC error	0
47	PARTITION3_MTCRC	Memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Attempt to access disabled window	0
65	AW1_INVALID_ACCESS	Attempt to access disabled window	0
66	AW2_INVALID_ACCESS	Attempt to access disabled window	0

Bits	Name	Usage	Default
67	AW3_INVALID_ACCESS	Attempt to access disabled window	0
68	AW4_INVALID_ACCESS	Attempt to access disabled window	0
69	AW5_INVALID_ACCESS	Attempt to access disabled window	0
70	AW6_INVALID_ACCESS	Attempt to access disabled window	0
71	AW7_INVALID_ACCESS	Attempt to access disabled window	0
72	AW8_INVALID_ACCESS	Attempt to access disabled window	0
73	AW9_INVALID_ACCESS	Attempt to access disabled window	0
74	AW10_INVALID_ACCESS	Attempt to access disabled window	0
75	AW11_INVALID_ACCESS	Attempt to access disabled window	0
76	AW12_INVALID_ACCESS	Attempt to access disabled window	0
77	AW13_INVALID_ACCESS	Attempt to access disabled window	0
78	AW14_INVALID_ACCESS	Attempt to access disabled window	0
79	AW15_INVALID_ACCESS	Attempt to access disabled window	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#) layout

Field AXI_A_PARITY

Error on bus AXI-A.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AXI_A_PARITY (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_A_PARITY values

Field AXI_B_PARITY

Error on bus AXI-B.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AXI_B_PARITY (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_B_PARITY values

Field AXI_C_PARITY

Error on bus AXI-C.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AXI_C_PARITY (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error

1	Error
---	-------

Field AXI_C_PARITY values

Field GENERAL_HARDWARE

General hardware error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Parity error on bus slice 0.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_PARITY values

Field SLICE1_PARITY

Parity error on bus slice 1.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_PARITY values

Field SLICE2_PARITY

Parity error on bus slice 2.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE2_PARITY (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_PARITY values

Field SLICE3_PARITY

Parity error on bus slice 3.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE3_PARITY (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_PARITY values

Field SLICE4_PARITY

Parity error on bus slice 4.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE4_PARITY (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_PARITY values

Field SLICE5_PARITY

Parity error on bus slice 5.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE5_PARITY (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_PARITY values

Field SLICE6_PARITY

Parity error on bus slice 6.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE6_PARITY (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_PARITY values

Field SLICE7_PARITY

Parity error on bus slice 7.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE7_PARITY (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Isolation error on bus slice 0.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE0_ISOLATION (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Isolation error on bus slice 1.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Isolation error on bus slice 2.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Isolation error on bus slice 3.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Isolation error on bus slice 4.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Isolation error on bus slice 5.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Isolation error on bus slice 6.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Isolation error on bus slice 7.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_ISOLATION values

Field SLICE_BIST

Slice BIST error.

[PTM_SYSTEM.PTM_IRQ_RAWSTAT](#).SLICE_BIST is stored in bits [27:20] and is a 8-bit bit set. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE_BIST values

Field INVALID_BUS

Enable invalid bus error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Watchdog error for group 0.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Watchdog error for group 1.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Watchdog error for group 2.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Watchdog error for group 3.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_MTCRC values

Field AW0_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW0_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW0_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_DEFERRED_ERROR_IRQ_STATUS.AW15_INVALID_ACCESS](#) (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW15_INVALID_ACCESS values

PTM_IRQ_INJECTION (0x0088 - 0x0090)

Inject system interrupts (before masking).

Inject interrupts to test interrupt handlers are correctly configured. This register is reset by PTM_GLOBAL_RESET but not by PTM_RECOVERY_RESET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	AXI_A_PARITY	Error on bus AXI-A	0
1	AXI_B_PARITY	Error on bus AXI-B	0
2	AXI_C_PARITY	Error on bus AXI-C	0
3	GENERAL_HARDWARE	General hardware error	0
4	SLICE0_PARITY	Parity error on bus slice 0	0
5	SLICE1_PARITY	Parity error on bus slice 1	0
6	SLICE2_PARITY	Parity error on bus slice 2	0
7	SLICE3_PARITY	Parity error on bus slice 3	0
8	SLICE4_PARITY	Parity error on bus slice 4	0
9	SLICE5_PARITY	Parity error on bus slice 5	0
10	SLICE6_PARITY	Parity error on bus slice 6	0
11	SLICE7_PARITY	Parity error on bus slice 7	0
12	SLICE0_ISOLATION	Isolation error on bus slice 0	0
13	SLICE1_ISOLATION	Isolation error on bus slice 1	0
14	SLICE2_ISOLATION	Isolation error on bus slice 2	0
15	SLICE3_ISOLATION	Isolation error on bus slice 3	0
16	SLICE4_ISOLATION	Isolation error on bus slice 4	0
17	SLICE5_ISOLATION	Isolation error on bus slice 5	0
18	SLICE6_ISOLATION	Isolation error on bus slice 6	0
19	SLICE7_ISOLATION	Isolation error on bus slice 7	0
28	INVALID_BUS	Enable invalid bus error	0
31:21	Reserved (zero)	-	-
32	GROUP0_WATCHDOG	Watchdog error for group 0	0
33	GROUP1_WATCHDOG	Watchdog error for group 1	0
34	GROUP2_WATCHDOG	Watchdog error for group 2	0
35	GROUP3_WATCHDOG	Watchdog error for group 3	0

Bits	Name	Usage	Default
36	PARTITION0_LOCKED_ACCESS	Attempt to access a locked partition register	0
37	PARTITION1_LOCKED_ACCESS	Attempt to access a locked partition register	0
38	PARTITION2_LOCKED_ACCESS	Attempt to access a locked partition register	0
39	PARTITION3_LOCKED_ACCESS	Attempt to access a locked partition register	0
40	PARTITION0_LOCKED_BIST	Attempt to access a locked BIST controller	0
41	PARTITION1_LOCKED_BIST	Attempt to access a locked BIST controller	0
42	PARTITION2_LOCKED_BIST	Attempt to access a locked BIST controller	0
43	PARTITION3_LOCKED_BIST	Attempt to access a locked BIST controller	0
44	PARTITION0_MTCRC	Memory transaction CRC error	0
45	PARTITION1_MTCRC	Memory transaction CRC error	0
46	PARTITION2_MTCRC	Memory transaction CRC error	0
47	PARTITION3_MTCRC	Memory transaction CRC error	0
63:48	Reserved (zero)	-	-
64	AW0_INVALID_ACCESS	Attempt to access disabled window	0
65	AW1_INVALID_ACCESS	Attempt to access disabled window	0
66	AW2_INVALID_ACCESS	Attempt to access disabled window	0
67	AW3_INVALID_ACCESS	Attempt to access disabled window	0
68	AW4_INVALID_ACCESS	Attempt to access disabled window	0
69	AW5_INVALID_ACCESS	Attempt to access disabled window	0
70	AW6_INVALID_ACCESS	Attempt to access disabled window	0
71	AW7_INVALID_ACCESS	Attempt to access disabled window	0
72	AW8_INVALID_ACCESS	Attempt to access disabled window	0
73	AW9_INVALID_ACCESS	Attempt to access disabled window	0
74	AW10_INVALID_ACCESS	Attempt to access disabled window	0
75	AW11_INVALID_ACCESS	Attempt to access disabled window	0
76	AW12_INVALID_ACCESS	Attempt to access disabled window	0
77	AW13_INVALID_ACCESS	Attempt to access disabled window	0
78	AW14_INVALID_ACCESS	Attempt to access disabled window	0
79	AW15_INVALID_ACCESS	Attempt to access disabled window	0
95:80	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_IRQ_INJECTION](#) layout

Field AXI_A_PARITY

Error on bus AXI-A.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AXI_A_PARITY (AP) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_A_PARITY values

Field AXI_B_PARITY

Error on bus AXI-B.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AXI_B_PARITY (BP) is stored in bit [1] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_B_PARITY values

Field AXI_C_PARITY

Error on bus AXI-C.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AXI_C_PARITY (CP) is stored in bit [2] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AXI_C_PARITY values

Field GENERAL_HARDWARE

General hardware error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).GENERAL_HARDWARE (GH) is stored in bit [3] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GENERAL_HARDWARE values

Field SLICE0_PARITY

Parity error on bus slice 0.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE0_PARITY (SP0) is stored in bit [4] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_PARITY values

Field SLICE1_PARITY

Parity error on bus slice 1.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE1_PARITY (SP1) is stored in bit [5] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field SLICE1_PARITY values

Field SLICE2_PARITY

Parity error on bus slice 2.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE2_PARITY (SP2) is stored in bit [6] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_PARITY values

Field SLICE3_PARITY

Parity error on bus slice 3.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE3_PARITY (SP3) is stored in bit [7] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_PARITY values

Field SLICE4_PARITY

Parity error on bus slice 4.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE4_PARITY (SP4) is stored in bit [8] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_PARITY values

Field SLICE5_PARITY

Parity error on bus slice 5.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE5_PARITY (SP5) is stored in bit [9] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_PARITY values

Field SLICE6_PARITY

Parity error on bus slice 6.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE6_PARITY (SP6) is stored in bit [10] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_PARITY values

Field SLICE7_PARITY

Parity error on bus slice 7.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE7_PARITY (SP7) is stored in bit [11] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_PARITY values

Field SLICE0_ISOLATION

Isolation error on bus slice 0.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE0_ISOLATION (SI0) is stored in bit [12] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE0_ISOLATION values

Field SLICE1_ISOLATION

Isolation error on bus slice 1.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE1_ISOLATION (SI1) is stored in bit [13] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE1_ISOLATION values

Field SLICE2_ISOLATION

Isolation error on bus slice 2.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE2_ISOLATION (SI2) is stored in bit [14] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE2_ISOLATION values

Field SLICE3_ISOLATION

Isolation error on bus slice 3.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE3_ISOLATION (SI3) is stored in bit [15] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE3_ISOLATION values

Field SLICE4_ISOLATION

Isolation error on bus slice 4.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE4_ISOLATION (SI4) is stored in bit [16] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE4_ISOLATION values

Field SLICE5_ISOLATION

Isolation error on bus slice 5.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE5_ISOLATION (SI5) is stored in bit [17] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE5_ISOLATION values

Field SLICE6_ISOLATION

Isolation error on bus slice 6.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE6_ISOLATION (SI6) is stored in bit [18] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE6_ISOLATION values

Field SLICE7_ISOLATION

Isolation error on bus slice 7.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).SLICE7_ISOLATION (SI7) is stored in bit [19] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field SLICE7_ISOLATION values

Field INVALID_BUS

Enable invalid bus error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).INVALID_BUS (IB) is stored in bit [28] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field INVALID_BUS values

Field GROUP0_WATCHDOG

Watchdog error for group 0.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).GROUP0_WATCHDOG (GW0) is stored in bit [32] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP0_WATCHDOG values

Field GROUP1_WATCHDOG

Watchdog error for group 1.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).GROUP1_WATCHDOG (GW1) is stored in bit [33] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP1_WATCHDOG values

Field GROUP2_WATCHDOG

Watchdog error for group 2.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).GROUP2_WATCHDOG (GW2) is stored in bit [34] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP2_WATCHDOG values

Field GROUP3_WATCHDOG

Watchdog error for group 3.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).GROUP3_WATCHDOG (GW3) is stored in bit [35] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field GROUP3_WATCHDOG values

Field PARTITION0_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION0_LOCKED_ACCESS (LA0) is stored in bit [36] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_ACCESS values

Field PARTITION1_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION1_LOCKED_ACCESS (LA1) is stored in bit [37] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_ACCESS values

Field PARTITION2_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION2_LOCKED_ACCESS (LA2) is stored in bit [38] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_ACCESS values

Field PARTITION3_LOCKED_ACCESS

Attempt to access a locked partition register.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION3_LOCKED_ACCESS (LA3) is stored in bit [39] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field PARTITION3_LOCKED_ACCESS values

Field PARTITION0_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION0_LOCKED_BIST (LB0) is stored in bit [40] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_LOCKED_BIST values

Field PARTITION1_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION1_LOCKED_BIST (LB1) is stored in bit [41] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_LOCKED_BIST values

Field PARTITION2_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION2_LOCKED_BIST (LB2) is stored in bit [42] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_LOCKED_BIST values

Field PARTITION3_LOCKED_BIST

Attempt to access a locked BIST controller.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION3_LOCKED_BIST (LB3) is stored in bit [43] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_LOCKED_BIST values

Field PARTITION0_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION0_MTCRC (MT0) is stored in bit [44] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION0_MTCRC values

Field PARTITION1_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION1_MTCRC (MT1) is stored in bit [45] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION1_MTCRC values

Field PARTITION2_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION2_MTCRC (MT2) is stored in bit [46] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION2_MTCRC values

Field PARTITION3_MTCRC

Memory transaction CRC error.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).PARTITION3_MTCRC (MT3) is stored in bit [47] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field PARTITION3_MTCRC values

Field AWO_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AWO_INVALID_ACCESS (IA0) is stored in bit [64] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW0_INVALID_ACCESS values

Field AW1_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW1_INVALID_ACCESS (IA1) is stored in bit [65] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW1_INVALID_ACCESS values

Field AW2_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW2_INVALID_ACCESS (IA2) is stored in bit [66] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW2_INVALID_ACCESS values

Field AW3_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW3_INVALID_ACCESS (IA3) is stored in bit [67] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW3_INVALID_ACCESS values

Field AW4_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW4_INVALID_ACCESS (IA4) is stored in bit [68] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW4_INVALID_ACCESS values

Field AW5_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW5_INVALID_ACCESS (IA5) is stored in bit [69] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW5_INVALID_ACCESS values

Field AW6_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW6_INVALID_ACCESS (IA6) is stored in bit [70] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW6_INVALID_ACCESS values

Field AW7_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW7_INVALID_ACCESS (IA7) is stored in bit [71] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW7_INVALID_ACCESS values

Field AW8_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW8_INVALID_ACCESS (IA8) is stored in bit [72] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW8_INVALID_ACCESS values

Field AW9_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW9_INVALID_ACCESS (IA9) is stored in bit [73] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW9_INVALID_ACCESS values

Field AW10_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW10_INVALID_ACCESS (IA10) is stored in bit [74] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW10_INVALID_ACCESS values

Field AW11_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW11_INVALID_ACCESS (IA11) is stored in bit [75] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW11_INVALID_ACCESS values

Field AW12_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW12_INVALID_ACCESS (IA12) is stored in bit [76] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW12_INVALID_ACCESS values

Field AW13_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW13_INVALID_ACCESS (IA13) is stored in bit [77] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW13_INVALID_ACCESS values

Field AW14_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW14_INVALID_ACCESS (IA14) is stored in bit [78] and is a 1-bit flag. Its default value is 0.

Value	Meaning
-------	---------

0 (default)	No error
1	Error

Field AW14_INVALID_ACCESS values

Field AW15_INVALID_ACCESS

Attempt to access disabled window.

[PTM_SYSTEM.PTM_IRQ_INJECTION](#).AW15_INVALID_ACCESS (IA15) is stored in bit [79] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	No error
1	Error

Field AW15_INVALID_ACCESS values

PTM_ERROR_FINGER_PRINT (0x0094)

Information indicating the cause of an error reported by an interrupt.

Used to indicate the cause of an error reported by an interrupt. The state of this register is preserved until the valid bit is cleared; errors arising while the valid bit is set do not record their cause. This register is reset by PTM_GLOBAL_RESET but not by PTM_RECOVERY_RESET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
0	VALID	Register holds valid data	0
31:1	FINGER_PRINT	Indicates cause of error including source (format)	0

Register [PTM_SYSTEM.PTM_ERROR_FINGER_PRINT](#) layout

Field VALID

Register holds valid data.

[PTM_SYSTEM.PTM_ERROR_FINGER_PRINT](#).VALID (V) is stored in bit [0] and is a 1-bit flag. Its default value is 0.

Value	Meaning
0 (default)	Not valid
1	Valid

Field VALID values

Field FINGER_PRINT

Indicates cause of error including source (format).

[PTM_SYSTEM.PTM_ERROR_FINGER_PRINT](#).FINGER_PRINT (F) is stored in bits [31:1] and is a 31-bit unsigned integer. Its default value is 0.

PTM_GROUP_RESET_STATE (0x0098)

Reset state of each group.

Reports reset state of resources owned by each group

Access to this Register is restricted to mode [ro](#).

Bits	Name	Usage
3:0	RESET0	Reset state of group 0
7:4	RESET1	Reset state of group 1
11:8	RESET2	Reset state of group 2
15:12	RESET3	Reset state of group 3
31:16	Reserved (zero)	-

Register [PTM_SYSTEM.PTM_GROUP_RESET_STATE](#) layout

Field RESET0

Reset state of group 0.

[PTM_SYSTEM.PTM_GROUP_RESET_STATE.RESET0](#) (R0) is stored in bits [3:0] and is a 4-bit enumeration of type PTM_RESET_ENUM.

The field can contain the following values:

Value	Name	Meaning
0	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET0 values

Field RESET1

Reset state of group 1.

[PTM_SYSTEM.PTM_GROUP_RESET_STATE.RESET1](#) (R1) is stored in bits [7:4] and is a 4-bit enumeration of type PTM_RESET_ENUM.

The field can contain the following values:

Value	Name	Meaning
0	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET1 values

Field RESET2

Reset state of group 2.

[PTM_SYSTEM.PTM_GROUP_RESET_STATE.RESET2](#) (R2) is stored in bits [11:8] and is a 4-bit enumeration of type PTM_RESET_ENUM.

The field can contain the following values:

Value	Name	Meaning
0	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET2 values

Field RESET3

Reset state of group 3.

[PTM_SYSTEM.PTM_GROUP_RESET_STATE](#).RESET3 (R3) is stored in bits [15:12] and is a 4-bit enumeration of type PTM_RESET_ENUM.

The field can contain the following values:

Value	Name	Meaning
0	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET3 values

PTM_GROUP_RESET_SET (0x009C)

Reset for each group.

Used to reset resources owned groups in the event of an error. Once the reset has been initiated it cannot be canceled.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
3:0	RESET0	Reset group 0	NO_RESET
7:4	RESET1	Reset group 1	NO_RESET
11:8	RESET2	Reset group 2	NO_RESET
15:12	RESET3	Reset group 3	NO_RESET
31:16	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_GROUP_RESET_SET](#) layout

Field RESET0

Reset group 0.

[PTM_SYSTEM.PTM_GROUP_RESET_SET](#).RESET0 (R0) is stored in bits [3:0] and is a 4-bit enumeration of type PTM_RESET_ENUM. Its default value is NO_RESET.

The field can contain the following values:

Value	Name	Meaning
0 (default)	NO_RESET	No reset

Value	Name	Meaning
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET0 values

Field RESET1

Reset group 1.

[PTM_SYSTEM.PTM_GROUP_RESET_SET](#).RESET1 (R1) is stored in bits [7:4] and is a 4-bit enumeration of type PTM_RESET_ENUM. Its default value is NO_RESET.

The field can contain the following values:

Value	Name	Meaning
0 (default)	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET1 values

Field RESET2

Reset group 2.

[PTM_SYSTEM.PTM_GROUP_RESET_SET](#).RESET2 (R2) is stored in bits [11:8] and is a 4-bit enumeration of type PTM_RESET_ENUM. Its default value is NO_RESET.

The field can contain the following values:

Value	Name	Meaning
0 (default)	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET2 values

Field RESET3

Reset group 3.

[PTM_SYSTEM.PTM_GROUP_RESET_SET](#).RESET3 (R3) is stored in bits [15:12] and is a 4-bit enumeration of type PTM_RESET_ENUM. Its default value is NO_RESET.

The field can contain the following values:

Value	Name	Meaning
0 (default)	NO_RESET	No reset
1	SOFT_RESET	Soft reset
2	HARD_RESET	Hard reset
3	Reserved	-

Field RESET3 values

PTM_ERROR_RESPONSE (0x00A0)

Response to errors.

Response to different types of error on each interface. This register is reset by PTM_GLOBAL_RESET but not by PTM_RECOVERY_RESET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
1:0	DCLS	Response to DCLS error	PASSIVE
3:2	PARTITION0_CRC	Response to partition 0 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET)	DISABLED
5:4	PARTITION1_CRC	Response to partition 1 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET)	DISABLED
7:6	PARTITION2_CRC	Response to partition 2 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET)	DISABLED
9:8	PARTITION3_CRC	Response to partition 3 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET)	DISABLED
11:10	Q_CHANNEL	Response to error on the Q-channel	PASSIVE
13:12	CLOCK	Response to clock errors	PASSIVE
15:14	MISC	Response to error on miscellaneous signals	PASSIVE
31:16	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_ERROR_RESPONSE](#) layout

Field DCLS

Response to DCLS error.

[PTM_SYSTEM.PTM_ERROR_RESPONSE.DCLS](#) (D) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_2_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2..3	Reserved	-

Field DCLS values

Field PARTITION0_CRC

Response to partition 0 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET).

[PTM_SYSTEM.PTM_ERROR_RESPONSE.PARTITION0_CRC](#) (P0) is stored in bits [3:2] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_3_ENUM. Its default value is DISABLED.

The field can contain the following values:

Value	Name	Meaning
0 (default)	DISABLED	Errors are ignored

Value	Name	Meaning
1	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	Reserved	-

Field PARTITION0_CRC values

Field PARTITION1_CRC

Response to partition 1 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET).

[PTM_SYSTEM.PTM_ERROR_RESPONSE.PARTITION1_CRC](#) (P1) is stored in bits [5:4] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_3_ENUM. Its default value is DISABLED.

The field can contain the following values:

Value	Name	Meaning
0 (default)	DISABLED	Errors are ignored
1	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	Reserved	-

Field PARTITION1_CRC values

Field PARTITION2_CRC

Response to partition 2 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET).

[PTM_SYSTEM.PTM_ERROR_RESPONSE.PARTITION2_CRC](#) (P2) is stored in bits [7:6] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_3_ENUM. Its default value is DISABLED.

The field can contain the following values:

Value	Name	Meaning
0 (default)	DISABLED	Errors are ignored
1	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	Reserved	-

Field PARTITION2_CRC values

Field PARTITION3_CRC

Response to partition 3 CRC error (CRC check may be overridden by PARTITION_CONTROL.PTM_MTCRC_SET).

[PTM_SYSTEM.PTM_ERROR_RESPONSE.PARTITION3_CRC](#) (P3) is stored in bits [9:8] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_3_ENUM. Its default value is DISABLED.

The field can contain the following values:

Value	Name	Meaning
0 (default)	DISABLED	Errors are ignored
1	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	Reserved	-

Field PARTITION3_CRC values

Field Q_CHANNEL

Response to error on the Q-channel.

[PTM_SYSTEM.PTM_ERROR_RESPONSE.Q_CHANNEL](#) (Q) is stored in bits [11:10] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_3_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	Reserved	-

Field Q_CHANNEL values

Field CLOCK

Response to clock errors.

[PTM_SYSTEM.PTM_ERROR_RESPONSE.CLOCK](#) (C) is stored in bits [13:12] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_2_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2..3	Reserved	-

Field CLOCK values

Field MISC

Response to error on miscellaneous signals.

[PTM_SYSTEM.PTM_ERROR_RESPONSE.MISC](#) (M) is stored in bits [15:14] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_2_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2..3	Reserved	-

Field MISC values

PTM_PARITY_RESPONSE (0x00A4)

Response to interface errors.

Response to parity errors. This register is reset by PTM_GLOBAL_RESET but not by PTM_RECOVERY_RESET.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
1:0	AXI_A_PARITY	Response to parity error on AXI-A	PASSIVE
3:2	AXI_B_PARITY	Response to parity error on AXI-B	PASSIVE
5:4	AXI_C_PARITY	Response to parity error on AXI-C	PASSIVE
7:6	SLICE0_PARITY	Response to parity error on slice 0 DRAM interface	PASSIVE
9:8	SLICE1_PARITY	Response to parity error on slice 1 DRAM interface	PASSIVE
11:10	SLICE2_PARITY	Response to parity error on slice 2 DRAM interface	PASSIVE
13:12	SLICE3_PARITY	Response to parity error on slice 3 DRAM interface	PASSIVE
15:14	SLICE4_PARITY	Response to parity error on slice 4 DRAM interface	PASSIVE
17:16	SLICE5_PARITY	Response to parity error on slice 5 DRAM interface	PASSIVE
19:18	SLICE6_PARITY	Response to parity error on slice 6 DRAM interface	PASSIVE
21:20	SLICE7_PARITY	Response to parity error on slice 7 DRAM interface	PASSIVE
31:22	Reserved (zero)	-	-

Register [PTM_SYSTEM.PTM_PARITY_RESPONSE](#) layout

Field AXI_A_PARITY

Response to parity error on AXI-A.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.AXI_A_PARITY](#) (A) is stored in bits [1:0] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field AXI_A_PARITY values

Field AXI_B_PARITY

Response to parity error on AXI-B.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.AXI_B_PARITY](#) (B) is stored in bits [3:2] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored

Value	Name	Meaning
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field AXI_B_PARITY values

Field AXI_C_PARITY

Response to parity error on AXI-C.

[PTM_SYSTEM.PTM_PARITY_RESPONSE](#).AXI_C_PARITY (C) is stored in bits [5:4] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field AXI_C_PARITY values

Field SLICE0_PARITY

Response to parity error on slice 0 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE](#).SLICE0_PARITY (S0) is stored in bits [7:6] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE0_PARITY values

Field SLICE1_PARITY

Response to parity error on slice 1 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE](#).SLICE1_PARITY (S1) is stored in bits [9:8] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE1_PARITY values

Field SLICE2_PARITY

Response to parity error on slice 2 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE2_PARITY](#) (S2) is stored in bits [11:10] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE2_PARITY values

Field SLICE3_PARITY

Response to parity error on slice 3 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE3_PARITY](#) (S3) is stored in bits [13:12] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE3_PARITY values

Field SLICE4_PARITY

Response to parity error on slice 4 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE4_PARITY](#) (S4) is stored in bits [15:14] and is a 2-bit enumeration of type PTM_ERROR_RESPONSE_4_ENUM. Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE4_PARITY values

Field SLICE5_PARITY

Response to parity error on slice 5 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE5_PARITY](#) (S5) is stored in bits [17:16] and is a 2-bit enumeration of type [PTM_ERROR_RESPONSE_4_ENUM](#). Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE5_PARITY values

Field SLICE6_PARITY

Response to parity error on slice 6 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE6_PARITY](#) (S6) is stored in bits [19:18] and is a 2-bit enumeration of type [PTM_ERROR_RESPONSE_4_ENUM](#). Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE6_PARITY values

Field SLICE7_PARITY

Response to parity error on slice 7 DRAM interface.

[PTM_SYSTEM.PTM_PARITY_RESPONSE.SLICE7_PARITY](#) (S7) is stored in bits [21:20] and is a 2-bit enumeration of type [PTM_ERROR_RESPONSE_4_ENUM](#). Its default value is PASSIVE.

The field can contain the following values:

Value	Name	Meaning
0	DISABLED	Errors are ignored
1 (default)	PASSIVE	Errors are reported but not acted on
2	ACTIVE_1	Corrupt transactions are suppressed (writes discarded reads return zero)
3	ACTIVE_2	As ACTIVE_1 but transactions with corrupt protocol signals are terminated

Field SLICE7_PARITY values

PTM_AW0_STREAM_ID (0x1000)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM](#).PTM_AW0_STREAM_ID layout

Field **STREAM_ID**

[PTM_SYSTEM](#).[PTM_AW0_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW0_PROTECTED_STREAM_ID (0x1004)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM](#).PTM_AW0_PROTECTED_STREAM_ID layout

Field **STREAM_ID**

[PTM_SYSTEM](#).[PTM_AW0_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW1_STREAM_ID (0x1008)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM](#).PTM_AW1_STREAM_ID layout

Field **STREAM_ID**

[PTM_SYSTEM](#).[PTM_AW1_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW1_PROTECTED_STREAM_ID (0x100C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW1_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW1_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW2_STREAM_ID (0x1010)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW2_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW2_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW2_PROTECTED_STREAM_ID (0x1014)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW2_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW2_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW3_STREAM_ID (0x1018)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW3_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW3_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW3_PROTECTED_STREAM_ID (0x101C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW3_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW3_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW4_STREAM_ID (0x1020)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW4_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW4_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW4_PROTECTED_STREAM_ID (0x1024)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW4_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW4_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW5_STREAM_ID (0x1028)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW5_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW5_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW5_PROTECTED_STREAM_ID (0x102C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW5_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW5_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW6_STREAM_ID (0x1030)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW6_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW6_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW6_PROTECTED_STREAM_ID (0x1034)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW6_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW6_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW7_STREAM_ID (0x1038)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW7_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW7_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW7_PROTECTED_STREAM_ID (0x103C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW7_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW7_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW8_STREAM_ID (0x1040)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW8_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW8_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW8_PROTECTED_STREAM_ID (0x1044)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW8_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW8_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW9_STREAM_ID (0x1048)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW9_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW9_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW9_PROTECTED_STREAM_ID (0x104C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW9_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW9_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW10_STREAM_ID (0x1050)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW10_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW10_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW10_PROTECTED_STREAM_ID (0x1054)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW10_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW10_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW11_STREAM_ID (0x1058)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW11_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW11_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW11_PROTECTED_STREAM_ID (0x105C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW11_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW11_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW12_STREAM_ID (0x1060)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW12_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW12_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW12_PROTECTED_STREAM_ID (0x1064)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW12_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW12_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW13_STREAM_ID (0x1068)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW13_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW13_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW13_PROTECTED_STREAM_ID (0x106C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW13_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW13_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW14_STREAM_ID (0x1070)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW14_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW14_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW14_PROTECTED_STREAM_ID (0x1074)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW14_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW14_PROTECTED_STREAM_ID.STREAM_ID](#) (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW15_STREAM_ID (0x1078)

Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW15_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW15_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

PTM_AW15_PROTECTED_STREAM_ID (0x107C)

Protected Stream ID.

Per-access window ID for memory transactions.

Access to this Register is restricted to mode [rw](#).

Bits	Name	Usage	Default
31:0	STREAM_ID	-	0

Register [PTM_SYSTEM.PTM_AW15_PROTECTED_STREAM_ID](#) layout

Field **STREAM_ID**

[PTM_SYSTEM.PTM_AW15_PROTECTED_STREAM_ID](#).STREAM_ID (ID) is stored in bits [31:0] and is a 32-bit unsigned integer. Its default value is 0.

C.1.10 Access states

There are several possible access states that the registers can have. These access states affect the access from the GPU, application processor, and the *MicroController Unit* (MCU) in the GPU.

const

This access type is used for configuration registers that are initialized at reset and are not changeable.

Table C-882: Permissions for access state const

Requester	Read	Write	Execute
Application processor	Allow	Fault	Fault
GPU	Allow	Fault	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

ro

This access type is used for unidirectional registers and buffers where the GPU is writing data for the application processor to read.

Table C-883: Permissions for access state ro

Requester	Read	Write	Execute
Application processor	Allow	Fault	Fault
GPU	Allow	Allow	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

ro_pri

This access type is used for unidirectional registers and buffers where the GPU is writing data for the application processor to read. In protected mode, the register retains its previous value and is not updated.

Attempting to access registers with this access state in protected mode does not cause a fault.

Table C-884: Permissions for access state ro_pri

Requester	Secure state	Read	Write	Execute
Application processor	Normal	Allow	Fault	Fault
Application processor	Protected	Allow	Fault	Fault
GPU	Normal	Allow	Allow	Fault
GPU	Protected	Allow	Ignore	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

ro_prr

This access type is used for unidirectional registers and buffers where the GPU is writing data for the application processor to read. In protected mode, only a restricted range of values is available.

Attempting to access registers with this access state in protected mode does not cause a fault.

Table C-885: Permissions for access state ro_prr

Requester	Secure state	Read	Write	Execute
Application processor	Normal	Allow	Fault	Fault
Application processor	Protected	Restricted	Fault	Fault
GPU	Any	Allow	Allow	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

ro_prz

This access type is used for unidirectional registers and buffers where the GPU is writing data for the application processor to read. In protected mode, application processor reads are not allowed.

Attempting to access registers with this access state in protected mode does not cause a fault.

Table C-886: Permissions for access state ro_prz

Requester	Secure state	Read	Write	Execute
Application processor	Normal	Allow	Fault	Fault
Application processor	Protected	Zero	Fault	Fault
GPU	Any	Allow	Allow	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

rw

This access type is used for unidirectional registers and buffers where the application processor is writing data for the GPU to read.

Table C-887: Permissions for access state rw

Requester	Read	Write	Execute
Application processor	Allow	Allow	Fault
GPU	Allow	Fault	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

rw_grw

This access type is used for bidirectional buffers, where both the application processor and GPU are permitted to read and write.

Table C-888: Permissions for access state rw_grw

Requester	Read	Write	Execute
Application processor	Allow	Allow	Fault
GPU	Allow	Allow	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

rw_prz

This access type is used for unidirectional registers and buffers where the application processor or MCU is writing data for the GPU to read. In protected mode, the register or field reads as constant-zero.

Table C-889: Permissions for access state rw_prz

Requester	Secure state	Read	Write
Application processor	Normal	Allow	Allow
GPU	Normal	Allow	Allow
MCU	Normal	Allow	Allow
Any	Protected	Zero	Allow

If the memory access does not match any of these cases, then the default behavior is `fault`.

wo

This access type is used for unidirectional registers and buffers where the application processor is writing data for the GPU to read.

The write-only property on the application processor enables side-effects to happen immediately without the requirement that the application processor can read them back.



This access type is Deprecated. It is simpler for virtualization if the application processor can read back the data.

Table C-890: Permissions for access state wo

Requester	Read	Write	Execute
Application processor	Fault	Allow	Fault
GPU	Allow	Fault	Fault

If the memory access does not match any of these cases, then the default behavior is `fault`.

Proprietary Notice

This document is protected by copyright and other related rights and the use or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm Limited ("Arm"). No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether the subject matter of this document infringes any third party patents.

The content of this document is informational only. Any solutions presented herein are subject to changing conditions, information, scope, and data. This document was produced using reasonable efforts based on information available as of the date of issue of this document. The scope of information in this document may exceed that which Arm is required to provide, and such additional information is merely intended to further assist the recipient and does not represent Arm's view of the scope of its obligations. You acknowledge and agree that you possess the necessary expertise in system security and functional safety and that you shall be solely responsible for compliance with all legal, regulatory, safety and security related requirements concerning your products, notwithstanding any information or support that may be provided by Arm herein. In addition, you are responsible for any applications which are used in conjunction with any Arm technology described in this document, and to minimize risks, adequate design and operating safeguards should be provided for by you.

This document may include technical inaccuracies or typographical errors. THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, any patents, copyrights, trade secrets, trademarks, or other rights.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Reference by Arm to any third party's products or services within this document is not an express or implied approval or endorsement of the use thereof.

This document consists solely of commercial items. You shall be responsible for ensuring that any permitted use, duplication, or disclosure of this document complies fully with any relevant

export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of this document shall prevail.

The validity, construction and performance of this notice shall be governed by English Law.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. Please follow Arm’s trademark usage guidelines at <https://www.arm.com/company/policies/trademarks>. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

PRE-1121-V1.0

Product and document information

Read the information in these sections to understand the release status of the product and documentation, and the conventions used in the Arm documents.

Product status

All products and Services provided by Arm require deliverables to be prepared and made available at different levels of completeness. The information in this document indicates the appropriate level of completeness for the associated deliverables.

Product completeness status

The information in this document is Final, that is for a developed product.

Revision history

These sections can help you understand how the document has changed over time.

Document release information

The Document history table gives the issue number and the released date for each released issue of this document.

Document history

Issue	Date	Confidentiality	Change
0305-20	17 April 2025	Non-Confidential	First release of version 3.5.
0304-19	4 November 2024	Non-Confidential	First release of version 3.4.
0303-18	30 July 2024	Non-Confidential	First release of version 3.3.
0302-17	31 May 2024	Non-Confidential	First release of version 3.2.
0301-16	27 March 2024	Non-Confidential	First release of version 3.1.
0300-15	2 February 2024	Non-Confidential	First release of version 3.0.
0208-14	16 August 2023	Confidential	First release of version 2.8.
0207-13	16 February 2023	Confidential	First release of version 2.7.
0206-12	23 September 2022	Confidential	First release of version 2.6.
0205-11	8 July 2022	Confidential	First release of version 2.5.

Issue	Date	Confidentiality	Change
0204-10	18 March 2022	Confidential	First release of version 2.4.
0203-09	15 October 2021	Confidential	First release of version 2.3.
0202-08	25 August 2021	Confidential	First release of version 2.2.
0201-07	2 July 2021	Confidential	First release of version 2.1.
0200-06	14 May 2021	Confidential	First release of version 2.0.
0103-05	25 February 2021	Confidential	First release of version 1.3.
0102-04	25 December 2020	Confidential	First release of version 1.2.
0101-03	25 September 2020	Confidential	First release of version 1.1.
0100-02	27 August 2020	Confidential	Second release of version 1.0.
0100-01	11 May 2020	Non-Confidential	First release of version 1.0.

The Change history tables describe the technical changes between released issues of this document in reverse order. Issue numbers match the revision history in [Document release information](#) on page 374.

Table 2: Differences between issue 0304-19 and 0305-20

Change	Location
Added a new section.	2.8.1 Porting information for other operating systems on page 55
Xen hardware hypervisor reorganized and renamed to accommodate new additional content.	B.1 Paravirtualization worked example on page 103
Added a new section.	B.2 Hardware virtualization worked example on page 105

Table 3: Differences between issue 0303-18 and 0304-19

Change	Location
Added a note about GPU power control.	2.1.5 Power management on page 19
Added the power management section.	2.3 Hardware virtualization on page 22
Updated the hardware virtualization section.	2.9 Power management in the virtualization reference stack on page 61
Corrected a register name from PTM_AW_ENABLE to PTM_AW_SET.	C.1.3 Register sub-page PTM_AW_GPU on page 147

Table 4: Differences between issue 0302-17 and 0303-18

Change	Location
Corrected the power module probe sequence.	2.8 Reference software stack for hardware virtualization on page 44

Table 5: Differences between issue 0301-16 and 0302-17

Change	Location
Added the paravirtualization time slice metrics.	2.2.1 Paravirtualization time slice metrics on page 21
Added the extra arbiter information to ARB_VM_GPU_STOP, ARB_VM_GPU_GRANTED, and ARB_VM_GPU_LOST.	3.2.1 Messages from arbiter to VM on page 88
Added the extra kbase information to VM_ARB_GPU_ACTIVE, VM_ARB_GPU_REQUEST, and VM_ARB_GPU_STOPPED.	3.2.2 Messages from VM to arbiter on page 90

Table 6: Differences between issue 0300-15 and 0301-16

Change	Location
Corrected the content by changing cl_arm_thread_limit_hint to cl_arm_scheduling_controls.	1.2.3 API usage and yielding on page 12

Table 7: Differences between issue 02-08-14 and 0300-15

Change	Location
Updated the reference paravirtualization software modules figure.	2.7 Reference software stack for paravirtualization on page 35

Table 8: Differences between issue 0207-13 and 0208-14

Change	Location
No technical changes.	-

Table 9: Differences between issue 0206-12 and 0207-13

Change	Location
Removed non-inclusive language.	Various
Correctly defined the OPP acronym.	2.9 Power management in the virtualization reference stack on page 61
Updated the partition manager system registers.	C.1.9 Register sub-page PTM_SYSTEM on page 245

Table 10: Differences between issue 0205-11 and 0206-12

Change	Location
Added steps of building models for virtualization.	2.10 Build reference software for virtualization on page 68
Updated the virtualization source path due to code relocation.	

Table 11: Differences between issue 0204-10 and 0205-11

Change	Location
Updated SDL-related information	2.13 Software security considerations for virtualization on page 81

Table 12: Differences between issue 0203-09 and 0204-10

Change	Location
Updated the information about Xen hypervisor.	2.10 Build reference software for virtualization on page 68

Table 13: Differences between issue 0202-08 and 0203-09

Change	Location
Added a new section.	1.2.4 Tiler queue size on page 14

Table 14: Differences between issue 0201-07 and 0202-08

Change	Location
Updated description.	1. Virtualization with GPUs on page 6, 2.1 Overview on page 16
Changed topic title from <i>GPU yield process</i> to <i>Cooperative time slicing</i> and updated description.	1.2 Cooperative time slicing on page 10
Changed text in figure.	1.2.1 Mali DDK data-flow on page 11
Added Xen versions and changed topic from <i>Build reference software for paravirtualization</i> to <i>Build reference software for virtualization</i> .	2.10 Build reference software for virtualization on page 68
Added a figure.	1.1.1 Paravirtualization with Mali GPUs on page 7
Expanded description and added a figure.	1.1.2.1 Access windows on page 9
Expanded description and added a figure	1.1.2.2 Hardware separation on page 9
Added topic and included GPU lost content.	1.2.2 GPU yield process on page 11
Changed topic title from <i>API usage and soft-stop</i> to <i>API usage and yielding</i> and removed GPU lost content.	1.2.3 API usage and yielding on page 12
Removed: <i>Supporting hardware for GPU virtualization</i> , <i>Memory isolation</i> , <i>Firewall</i> , <i>Soft stop</i> , and <i>Protected content and soft-stop</i> .	1. Virtualization with GPUs on page 6
Added FuSa statement.	2. Reference software stacks for virtualization on page 16
Added error messages for System.	2.8 Reference software stack for hardware virtualization on page 44
Added MALI_PARTITION_MANAGER.	2.10 Build reference software for virtualization on page 68

Table 15: Differences between issue 0200-06 and 0201-07

Change	Location
Added a new section.	2.5 Resource assignment with the partition manager on page 27
Added a new chapter.	C. Partition manager on page 114
Updated the steps for <ul style="list-style-type: none"> Build the reference code. Building the kernel in the tree. 	2.10 Build reference software for virtualization on page 68

Table 16: Differences between issue 0103-05 and 0200-06

Change	Location
Chapter updated.	1. Virtualization with GPUs on page 6
	3. Message protocol on page 85
	2. Reference software stacks for virtualization on page 16
	B. Xen hypervisor on page 102
	A. Limitations on virtualization in GPUs on page 100

Table 17: Differences between issue 0102-04 and 0103-05

Change	Location
Updated the section of Hypervisors.	Removed in version 2.0.

Table 18: Differences between issue 0101-03 and 0102-04

Change	Location
Updated the platform information.	2.10 Build reference software for virtualization on page 68.

Table 19: Differences between issue 0100-02 and 0101-03

Change	Location
Removed the section of Workaround for interrupt handling.	2.10 Build reference software for virtualization on page 68
Updated the steps of Build the reference code.	
Updated the GPU device tree node.	B. Xen hypervisor on page 102
Removed the content related with pausing the domain.	Various

Table 20: Differences between issue 0100-01 and 0100-02

Change	Location
Corrections for patch code of applying the Xen hypervisor manually.	2.10 Build reference software for virtualization on page 68

Table 21: Issue 0100-01

Change	Location
First release for version 1.0.	-

Conventions

The following subsections describe conventions used in Arm documents.

Glossary

The Arm Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the Arm Glossary for more information: developer.arm.com/glossary.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
<i>italic</i>	Citations.
bold	Interface elements, such as menu names. Terms in descriptive lists, where appropriate.

Convention	Use
monospace	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
monospace <u>underline</u>	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: <pre>MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2></pre>
SMALL CAPITALS	Terms that have specific technical meanings as defined in the <i>Arm® Glossary</i> . For example, IMPLEMENTATION DEFINED , IMPLEMENTATION SPECIFIC , UNKNOWN , and UNPREDICTABLE .



We recommend the following. If you do not follow these recommendations your system might not work.



Your system requires the following. If you do not follow these requirements your system will not work.



You are at risk of causing permanent damage to your system or your equipment, or of harming yourself.



This information is important and needs your attention.



This information might help you perform a task in an easier, better, or faster way.



This information reminds you of something important relating to the current content.

Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Arm documents are available on developer.arm.com/documentation.

Confidential documents are only available to licensees, when logged in. Each document link in the tables below provides direct access to the online version of the document.

Arm product resources	Document ID	Confidentiality
<i>Arm® GPU DDK Integration Manual</i>	109558	Confidential
<i>Arm® Mali™-G78AE GPU Configuration and Integration Manual</i>	102189	Confidential
<i>Arm® Mali™-G78AE GPU Technical Overview</i>	102188	Confidential
<i>Arm® Mali™-G78AE GPU Technical Reference Manual</i>	102187	Confidential