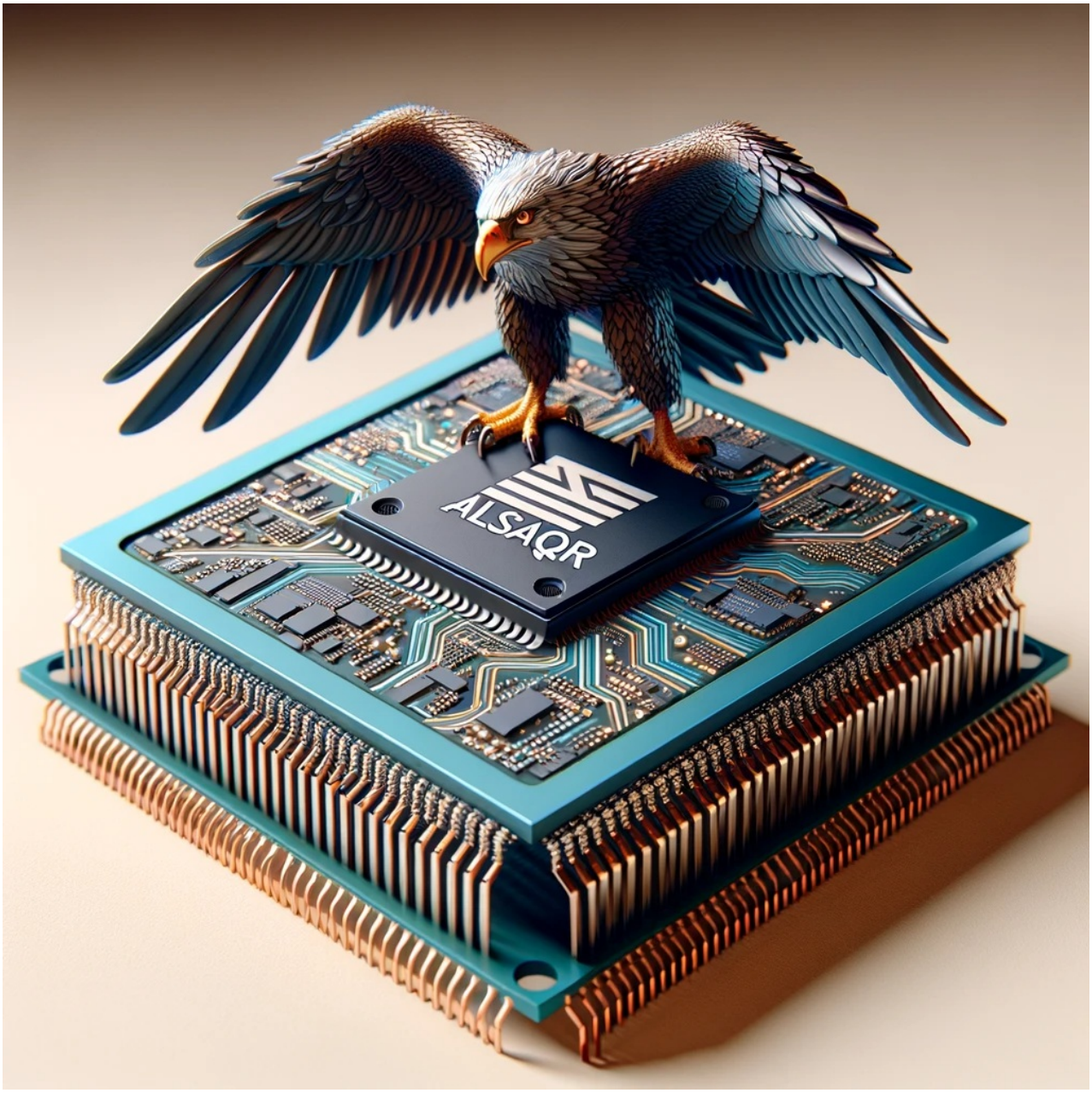

AISaqr - A PARALLEL ULTRA-LOW POWER PROCESSOR FOR DRONE APPLICATIONS

AISaqr Hardware Reference Manual

Version 1.0.0

This document is preliminary and subject to change

Micrel Lab, University of Bologna ,Italy

Table of Contents

1 Overview

AlSaqr project is a custom System-on-Chip based on the Parallel Ultra-Low-Power Processor (PULP) architecture from University of Bologna Energy Efficient Embedded System Lab and ETH Zürich Integrated Systems Lab. The AlSaqr implementation extends the architecture to provide the necessary systems interfaces, capabilities and securety features required by drone and nano-drone application.

This document introduces an advanced SoC design based on RISC-V ISA, meticulously engineered for drone and nano-drone applications, marking a significant milestone in this rapidly progressing field.

This project has been conceived and executed with dual objectives: aligning with the current apex of UAV technology while pushing the boundaries in performances and security features, a critical aspect often overlooked in rapid technological advancements.

2 Features

- - 2 high performance CVA6 64-bit RISC-V core
  - 8 CV32E40P 32-bit RISC-V cores
- Security Subsystem
- Memories:
  - L1 Memory (256 KB) shared by all the cores in Cluster (0 wait state memory access)
  - L2 Memory (32KB) for all the cores
  - Last Level Cache (128KB)
  - L3 external HyperFlash or HyperRAM Memory connnected to the HyperBus Interface
  - ROM (64KB)
- Clock, reset and supply management
  - 1 FLL with 4 programmable outputs
    - Out 0: CVA6
    - Out 1: SoC
    - Out 2: Peripherals
    - Out 3: PULP Cluster
  - 1 x 32.768kHz RTC
  - Single 32.768kHz crystal for RTC and FLLs
- Debug Mode
  - JTAG interface
- DMA
  - A multi-channel 1D/2D cluster-DMA controls the transactions between the L2 Memory and L1 Memory
  - A smart, lightweight and completely autonomous uDMA capable of handling complex I/O scheme
- 2 x Camera Parallel Interfaces CPI
  - 8 bits interface
  - HSYNC, VSYNC, PCLK
- Communication Interfaces
  - 6 x I2C Master
  - 11 x SPI Master
  - 1 x QSPI
  - 3 x UART
  - 4 x USART
  - 2 x SDIO
  - 2 x CAN
  - 1 x HYPERBUS
  - 1 x Ethernet

3 Introduction to the Al Saqr application architecture

AlSaqr SoC is designed to cater to the compatibility with FMUv6X specification offering intricate and demanding requirements of drone and nano-drone systems. These systems require a blend of processing power, energy efficiency, and real-time data handling capabilities, all within the constraints of minimal weight and size – attributes that are at the core of our SoC design. The goal of AlSaqr project is to replace the Pixhawk FMUv6X in two different use cases:

- Nano Drones, as a whole flight computer
- Micro & Standard Drones as a flight controller, placed aside to a Mission Computer

Drones outperform traditional methods and heavy machinery by being 10 times more efficient, 100 times more cost-effective, and significantly safer across various applications:

- For public safety, drones offer easier deployment and scalability compared to helicopters.
- In tower inspections, drones eliminate the need for dangerous climbing, enhancing inspector safety.
- During bridge inspections, drones cause minimal disruption compared to the use of snooper trucks. As the era of manually operated drones reaches its zenith, we are transitioning to a new phase dominated by software-controlled aircraft, heralded as the “Age of AI-driven autonomy.” This advancement will see AI-powered unmanned aerial vehicles (UAVs) navigating and making decisions autonomously based on sensor inputs, primarily through cameras. To achieve this, drones will utilize a variety of algorithms including:
- Nonlinear least-squares optimization
    - Visual odometry
    - Simultaneous localization and mapping (SLAM)
    - System identification
    - Model predictive control
- Deep learning
- Motion planning
- Low-level image processing
- Geometric computation

4 Verification

The SoC verification has been performend on the VCU118 followingthese steps . The peripherals have been validated using the open source FMC peripheral boards, a set of PCB designs specifically created for validating peripherals on various FPGA platforms enhanced with the FPGA Mezzanine Card (FMC) standard.

Please refer to this repository for source files and schematics.

5 Host Domain

The AlSaqr SoC is designed around a 64-bit architecture, significantly bolstered by two cva6 cores that form the central computing engine. The architecture incorporates memories and peripherals, notably featuring a dedicated HyperRAM controller that facilitates access to off-chip L3 memory, thereby extending the storage capabilities and ensuring ample memory bandwidth for data-intensive operations.

At the core of AlSaqr's design lies the AXI4 interconnect, a high-bandwidth, low-latency communication framework that adeptly connects the cva6 cores with the Cluster domain, the Secure Subsystem domain and the HyperRAM controller. To enhance the efficiency of data transfers and minimize the need for frequent accesses to the external L3 memory, a Last Level Cache (LLC) is strategically positioned between the HyperRAM memory controller and the AXI4 interconnect. This LLC acts as a buffer, storing frequently accessed data close to the processing cores, thereby significantly reducing latency and improving overall system performance.

Further augmenting the architecture is the Advanced Peripheral Bus (APB) subsystem, tasked with the efficient management of peripheral components. This includes the uDMA engine for streamlined data movement between peripherals and memories, reducing CPU load, and GPIOs for flexible input/output options, enabling interactions with a wide array of external devices and sensors. Additionally, AlSaqr-specific registers, accessible via the APB subsystem, provide detailed control over the SoC's functionalities, enabling tailored performance adjustments to meet specific application needs.

Thanks to the Platform Level Interrupt Controller PLIC, AlSaqr maneges up to 150 interrupts and exceptions. The PLIC plays a significant role in ensuring that the processor can handle various events, such as hardware interrupts or software exceptions, efficiently.

The following figure describes the main functional blocks of AlSaqr :

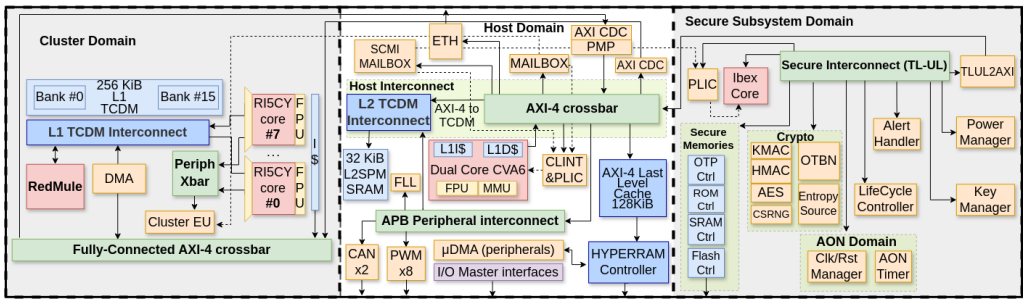

Figure 1. AlSaqr SoC

6 Clock Domains

The AlSaqr chip employs a sophisticated clocking system with four distinct clock domains, each serving different components of the chip's architecture. These clock domains are generated by an FLL capable of producing four programmable clock sources for the following domains:

- Dual-Core Domain – 800 MHz The Dual-Core Domain handles the clocking requirements for the dual-core processor units, facilitating their synchronous operation and coordination.
- SoC Domain – 350 MHz This clock domain governs the System-on-Chip (SoC) components, which encompass the core functionalities and operations of the chip.
- Peripheral Domain – 320 MHz The Peripheral Domain is responsible for clocking peripheral devices and interfaces handled by the uDMA core.
- Cluster Domain – 500 MHz This domain manages the clocking needs of clusters of related components or subsystems within the chip architecture, including RedMule accelerator.
- Security Subsystem Domain – 400MHz The Security Subsystem of AlSaqr features its own clock domain, which can be dynamically selected from among the four main sources using a memory-mapped register. This flexibility allows for tailored clocking configurations to suit the specific requirements and security considerations of the subsystem. Furthermore, the clock source selected for the security subsystem or any of the primary domains can be further divided by a programmable memory-mapped register. This capability enables fine-grained control over the clock frequencies within each domain, allowing for optimization of power consumption, performance, and synchronization.

In addition to the internal clock sources generated by the FLL, AlSaqr SoC offers the flexibility to bypass these sources. This feature can be easily initiated by driving LOW or HIGH a dedicated input pin of the AlSaqr chip.

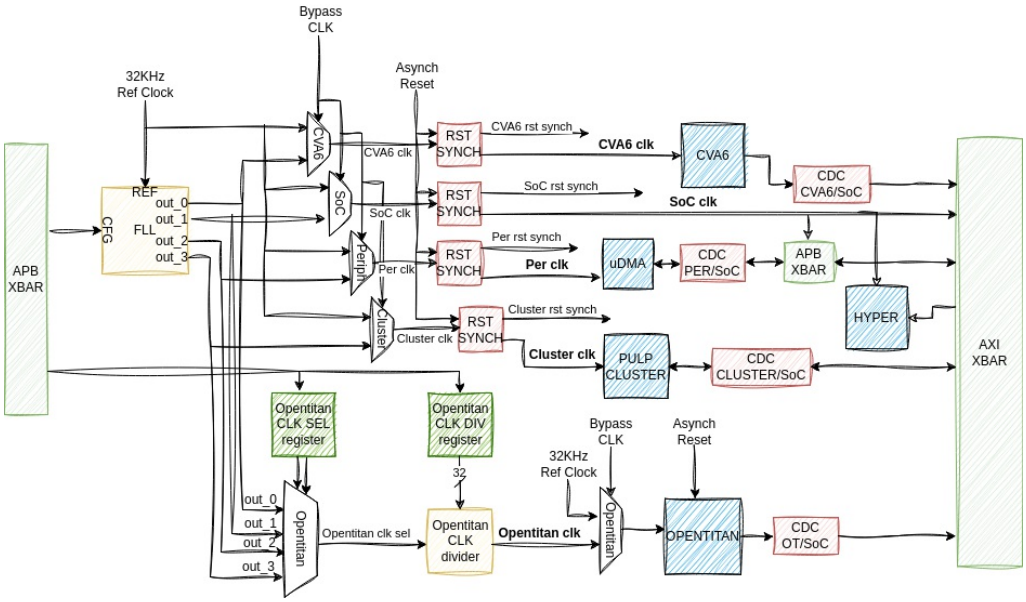

Figure 1. Clock Tree

6.1 CVA6 Dual Core

The CVA6 core implements the RISC-V instruction set architecture (ISA), marking it as a versatile and powerful choice for a wide range of

computing applications. CVA6 is a 6-stage, single issue, in-order CPU which implements I, M, A and C extensions. The design includes virtual memory and privileged mode operations, essential features for running Linux, as they enable efficient memory management and secure execution of privileged tasks.

The dual-core implementation of the CVA6 is designed around the AXI ACE protocol which is an extension of the AXI4 protocol, enhances support for hardware-coherent caches within a system. The cache coherency protocol used in this implementation is MOESI (Modified, Owned, Exclusive, Shared, Invalid). This protocol is an extension of the MESI protocol, providing an additional state to optimize the management of shared data and reduce the bandwidth required for cache coherence traffic.

This setup allows for the effective sharing and coherency of data across the cores, ensuring that each core has timely access to the correct data, which is critical for the performance and correctness of multi-threaded and multi-processed applications.

6.2 Core Local Interrupt Controller (CLINT)

The standard RISC-V platform-level interrupt controller (PLIC) provides centralized interrupt prioritization and routing for shared platform-level interrupts, and sends only a single external interrupt signal per privilege mode (meip/seip/ueip) to each hart. The PLIC multiplexes various device interrupts onto the external interrupt lines of Hart contexts, with hardware support for interrupt priorities. PLIC supports up-to 1023 interrupts (0 is reserved) and 15872 contexts, but the actual number of interrupts and context depends on the PLIC implementation. However, the implement must adhere to the offset of each register within the PLIC operation parameters. Further details can be found at this page

6.3 Platform Level Interrupt Controller - PLIC

The RISC-V PLIC (Platform-Level Interrupt Controller) is a crucial component in AlSaqr SoC. Its primary functions include the management of 150 interrupt sources and the prioritization of these interrupts based on their significance. The PLIC is designed to ensure that the processor can efficiently handle external events, such as hardware interrupts or software exceptions.

Within the PLIC, interrupt sources are mapped to specific priority levels, allowing for orderly handling. In AlSaqr the PLIC supports four priority levels (machine mode, supervisor mode, user mode), and higher-priority interrupts are serviced before lower-priority ones. This prioritization is a fundamental aspect of its operation. When an interrupt occurs, the PLIC identifies the source of the interrupt and routes it to the appropriate interrupt service routine (ISR) or handler. This process is known as interrupt vectoring. The PLIC also allows for the enabling and disabling of individual interrupt sources, giving flexibility in interrupt management. To interact with the PLIC, a processor typically uses memory-mapped registers or software instructions. These registers control the enabling/disabling of interrupts, setting priorities, and claiming/completing interrupts. Depending on the privilege level, the PLIC handles interrupts accordingly.

Further documentation regarding the RISC-V PLIC can be found at this page

6.4 Memory Subsystem

The memory subsystem described above is a three-level memory hierarchy composed of L3, L2 and L1. The main memory of the architecture is the L3 HyperRAM or HyperFlash memory, which is located off-chip and has a variable size (8x8 - 64x8) MB. A 128KB of Last Level Cache (LLC) is designed to serve as a buffer between the L3 and the processor, providing faster access to frequently used data. It copes with the intrinsic high latency of the HyperBus protocol and delivers maximum performance. A subset of the LLC can be configured to act as a scratchpad memory (SPM).

The L2 memory, with a size of 32KB represents the memory for the host domain. The L2 is a SPM, on-chip and is directly accessible by the processors and the IO engine.

The L1 SPM, with a size of 256KB, is the memory for the cluster domain. It is also on-chip and is designed to provide even faster access to the cluster accelerator. The cluster DMA engine moved data between L2 and L1.

Overall, this memory subsystem is designed to ensure that frequently used data is stored in the fastest and most accessible memory closer to the processor, while less frequently used data is stored in larger and slower memory further away from the processor. This ensures that data can be accessed quickly by the processor, reducing memory access latency and improving overall performance.

6.4.1 HyperRAM Contrtoller

Communication with the L3 memory is managed by the HyperRAM controller, which implements an AXI4-compliant interface for the HyperBus protocol. Further details about this protocol can be found here.

The controller includes two internal PHYs and can be configured through registers mapped to the APB subsystem. It supports operation with either one or both PHYs enabled.

Multiple memories connected to the same bus are organized contiguously in the address map and individually selected through dedicated chip select (CS) signals. At runtime, the HyperRAM sizes can be provided to the controller, which then appropriately demultiplexes transactions. Both PHY interfaces support the same number of CS lines (four each). When utilizing two HyperBuses, memory pairs sharing the same chip select line are mapped in an interleaved manner, with each memory appearing as a distinct 16-bit-wide memory block.

The block diagram below illustrates the overall architecture. Note the two internal clock domains: clk_sys and clk_phy. Additionally, the controller receives an external clock (clk_rwds_in), which is internally delayed. The clocks clk_phy and clk_phy_90 differ in phase by 90 degrees. These two clocks can be generated either by shifting the input clock clk_phy_i using the internal clock delayer or by employing the ddr_clk module, which halves the frequency and produces four clocks shifted by 90 degrees relative to each other.

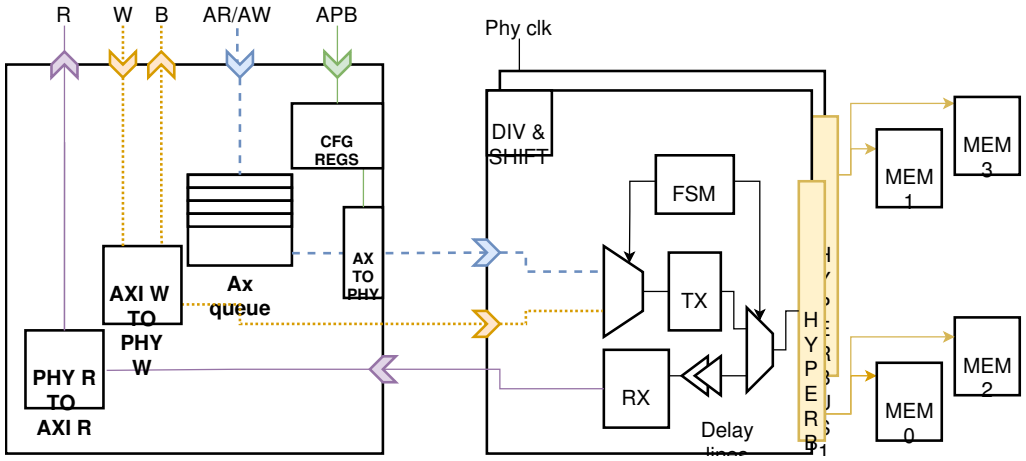

Figure 1. Block-diagram Hyperram controller.

6.4.1.1 Single PHY configuration for AlSaqr QFN100 chip

Here is an example to configure the Hyperram Controller to use only PHY0 targeting thefollowing S80KS5122 HyperRAM memories

Config Hyperram delay (memory bank dependent)
0x1A101000 0x7

Config Address Mask MSB (memory bank dependent)
0x1A101018 0x1B

Config Address space
0x1A10101C 0x0

Config PHY in use to 0
0x1A101020 0x0

Config which PHY to 0
0x1A101024 0x0

CS0 range
Start: 0x1A101028 0x80000000
End: 0x1A10102C 0x84000000

CS1 range
Start: 0x1A101030 0x84000000
End: 0x1A101034 0x88000000

CS2 range
Start: 0x1A101038 0x88000000
End: 0x1A10103C 0x8C000000

CS3 range
Start: 0x1A101040 0x8C000000
End: 0x1A101044 0x90000000

6.4.2 Last Level Cache

The Last Level Cache (LLC) is a 128KB, 8-way set associative cache. Each way being 256 sets, and each set containing 8 blocks of 64-bit data. The LLC features the write back (WB) policy, is capable of handling multiple outstanding transactions and implements the pseudo-random replacement policy. Each way can be configured to be used as SPM. The memory map of the cached and SPM regions can be configured at runtime via software.

To set the ways of the LLC in either cache or SPM mode you need to configure the CFG_SPM registers. As an example, configuring CFG_SPM_LOW to 0b00000101 will set way 0 and way 2 to SPM, and the rest to cache. It is also possible to manually flush LLC ways using the CFG_FLUSH and FLUSHED registers. For example, writing 0b00000101 to CFG_FLUSH_LOW will request flushing of ways 0 and 2. By monitoring the FLUSHED_LOW register it is possible to keep track of the status of such requests.

The SPM memory addresses (i.e. memory map of the SPM region) can be specified by configuring the LLC_SPM_ADDR_START register. The memory map will begin at the value specified in LLC_SPM_ADDR_START and will have size equal to Nx16KB, where $0 \leq N \leq 8$ is the number of ways set to SPM mode. If all 8 ways are set to SPM mode, the entire 128KB of LLC will be used as SPM.

The cached memory addresses (i.e. memory map of the cached region) can be specified by configuring the LLC_CACHE_ADDR_START

and LLC_CACHE_ADDR_END registers. As an example, setting LLC_CACHE_ADDR_START and LLC_CACHE_ADDR_END to 0x80000000 and 0x80800000 respectively, will cache this address region, unless all ways of the LLC are set to SPM mode. In general, when all ways are set to SPM, the LLC will bypass non-SPM memory addresses. The LLC will also bypass non-cached and non-SPM memory addresses, regardless of the configuration of each way.

The design uses the AXI4 protocol for it's data transfers. An AXI4 slave port handles transactions from the CPU side, whereas a master port faces the main memory side. It is fully configurable in size and cache dimensions and features a runtime configurable scratch-pad mode (SPM) for the different sets. Meaning that individual sets (also called ways) of the cache can be disabled for hardware managed caching and directly addressed when configured in this mode. This additional functionality reuses most of the logic. The cache features a transaction bypass for enabling direct accesses from the slave port to the master port. This bypass can be used when all of the sets of the cache are configured as SPM or when accessing address ranges outside the SPM and CACHE regions. The LLC further has user activated flushing capabilities, performance counters and an automatic build in self test during initialization for the tag SRAM storage. These functionalities are controlled over a dedicated AXI4 LITE configuration port.

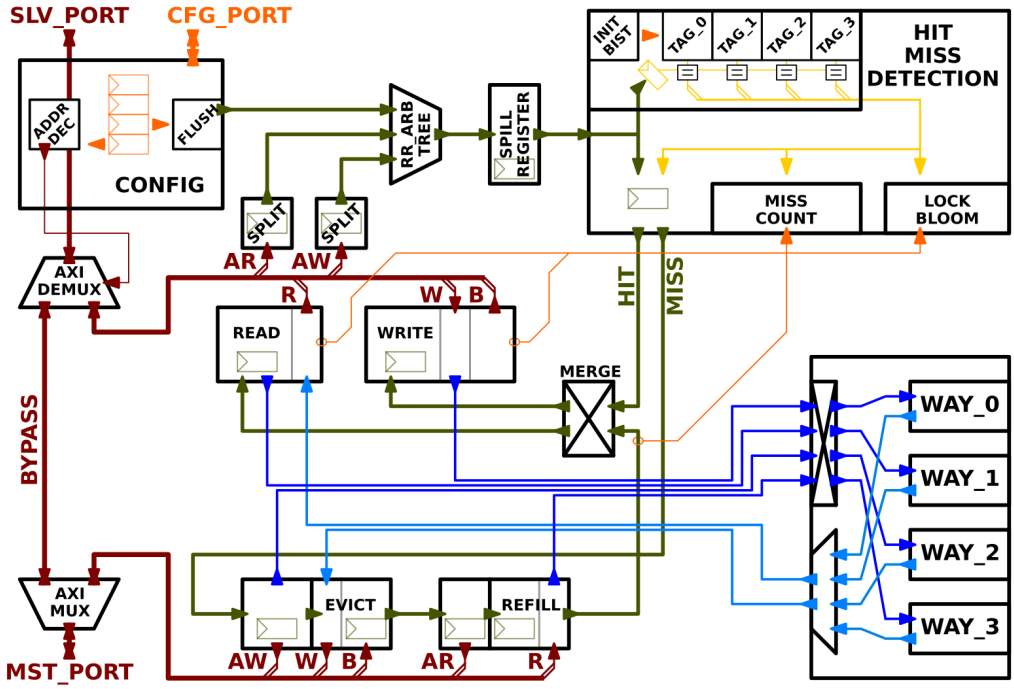

Figure 2. Block-diagram he Top Level of the LLC.

The LLC has a number of fixed parameters which have to be given during instantiation. The module features three main parameters, which determine the overall size and shape of the LLC. They are all of the type int unsigned.

- SetAssociativity: The set-associativity of the LLC. This parameter determines how many ways/sets will be instantiated. The minimum value is 1. The maximum value depends on the data width of the AXI LITE configuration port and should either be 32 or 64 to stay inside the protocol specification. The reason is that the SPM configuration register matches in width the data width of the LITE configuration port.
- NoLines: Specifies the number of lines in each way. This value has to be higher than two. The reason is that in the address at least one bit has to be mapped onto a cache-line index. This is a limitation of the *system verilog* language, which requires at least one bit wide fields inside of a struct. Further this value has to be a power of 2. This has to do with the requirement that the address mapping from the address onto the cache-line index has to be continuous.
- NoBlocks: Specifies the number of blocks inside a cache line. A block is defined to have the data width of the master port. Currently this also matches the data with of the slave port. The same value limitation as with the *NoLines* parameter applies. Fixing the minimum value to 2 and the overall value to be a power of 2.

There exists an additional top level parameter which is compromised of a struct defining the AXI parameters of the different ports. The definition can be found in axi_llc_pkg. Following table defines the fields of this struct which are all of typeint unsigned.

Name	Function
SlvPortIdWidth	AXI ID width of the slave port, facing the CPU side.
AddrWidthFull	AXI address width of both the slave and master port.
DataWidthFull	AXI data width of both the slave and the master port.
LitePortAddrWidth	AXI address width of the configuration LITE port.
LitePortDataWidth	AXI data width of the configuration LITE port. Has to be ether 32 or 64 bit to adhere to the AXI4 specification. Further limits the maximum set-associativity of the cache.

It is also required to provide the detailed AXI4 structs as parameters for the respective ports. The structs follow the naming scheme *port_xx*_chan_t. Where *port* stands for the respective port and have the values *slv*, *mst* and *lite*. In addition the respective request and response structs have to be given. The address rule struct from the common_cells addr_decode have to be specified for the FULL and LITE AXI4 ports as they are used internally to provide address mapping for the AXI transfers onto the different SPM and cache regions.

The overall size in bytes of the LLC in byte can be calculated with:

$$LlcSize = SetAssocitativity * NoLines * NoBlocks * \frac{DataWidthFull}{8}$$

Figure 3. Equation axi_llc size

The AXI4 protocol issues its transfers in a bursted fashion. The idea of the LLC is to use most of the control information provided by the protocol to implement the cache control in a decentralized way. To achieve that a data-flow based control scheme is chosen.

The premise being that an AXI transfer gets translated into a number of descriptors which then flow through a pipeline. Each descriptor maps the specific operation on a cache line and basically translates long bursts onto shorter ones which exactly map onto a cache line. For example when an AXI burst wants to write on three cache-lines the control beat gets translated into three descriptors which then flow through the pipeline.

It follows a brief example what happens to a write transfer when it accesses the cache. An AW beat is valid on the slave port of the LLC. Its address gets decoded in the configuration module, passes through the split unit and its first descriptor enters the spill register. In the next cycle a request gets issued to the tag-storage, which is compromised of one SRAM block per set of the cache. In the next cycle the hit or miss gets determined. The descriptor then goes either directly to the write unit if its a hit, or to the eviction/refill pipeline if it was a miss. This hit bypass allows AXI transaction which hit onto the cache to overtake ones that are in the miss pipeline. This has the advantage that a short write transaction for example from a CPU can overtake a long read transaction which could have been issued by a DMA for example. On a miss the descriptor enters the miss pipeline. If the cache-line was dirty it gets evicted by issuing a write request on the master port. Next the cache refills the line from main memory. Then the descriptor gets transferred into the write unit which then sends the W beats from the CPU towards the data storage. When all W beats are transferred, the write unit issues a response back to the CPU, thus ending the transfer.

Following table defines the struct which is used to describe a cache descriptor. Part of the descriptor uses directly types defined in axi_pkg. The other fields get defined when instantiating the design using the parameters from the top level.

Name	Type	Function
a_x_id	axi_slv_id_t	The AXI4 ID of the burst entering through the slave port of the design. It has the same with as the slave AXI ID.
a_x_addr	axi_addr_t	The address of the descriptor. However it gets defines, so that it maps to the corresponding cache line inside the split modules.
a_x_len	axi_pkg::len_t	AXI4 burst length field. Corresponds to the number of beats which map onto this descriptor. It gets set in the splitting unit which does the mapping onto the cache line.
a_x_size	axi_pkg::size_t	The AXI4 size field. This is important for the write and read unit to find the exact block and byte offset, where a beat from the slave port should go in the data storage.
a_x_burst	axi_pkg::burst_t	The burst type of the AXI4 transaction. This is important for the splitter unit as well as the read and write unit, as it determines the descriptor field a_x_addr.
a_x_lock	logic	The AXI4 lock signal, currently only passed further in the miss pipeline when the line gets evicted or refilled.
a_x_cache	axi_pkg::cache_t	The AXI4 cache signal. Currently the cache only supports write back mode.
a_x_prot	axi_pkg::prot_t	The AXI4 protection signal, currently only passed further in the miss pipeline.
x_resp	axi_pkg::resp_t	The AXI4 response signal. This also tells if we try to make un-allowed accesses onto address regions which are not currently mapped to ether SPM nor cache. When this signal gets set somewhere in the pipeline, all following modules will pass the descriptor along and absorb the corresponding beats on the ports.
x_last	logic	The AXI4 last flag. Here it determines, if the read or write unit sen back the response, when the last beat of the descriptor gets transferred on the slave port.
spm	logic	This field signals that the descriptor is of type SPM. It will not make a lookup in the hit/miss detection and utilize the hit bypass.
rw	logic	This field determines if the descriptor makes a write access 1'b1 or read access 1'b0.
way_ind	logic	The way indicator. Is a vector of width equal of the set-associativity setting of the LLC and decodes the index of the cache way where the descriptor should make an access.
evict	logic	The eviction flag. The descriptor missed and the line at the position was determined dirty by the detection. The evict unit will write back the dirty cache-line to the main memory.
evict_tag	logic	The eviction tag. As the field a_x_addr has the new tag in it, it is used to send back the right address to the main memory during eviction.
refill	logic	The refill flag. The descriptor will trigger a read transaction to the main memory, refilling the cache-line.
flush	logic	The flush flag. This only gets set when a way should be flushed. It gets only set by descriptors coming from the configuration module.

6.4.2.0.1 Transaction Split and SPM

This section elaborates in more detail on how the LLC determines a SPM access and how an AXI transaction gets split onto the descriptors defining the cache-line operation.

The modules in axi_llc_top uses the address decoding module from common_cells. When instantiated the LLC calculates the address map for the SPM region dependent from the address port called spm_start_addr_i. The port has the same width as the corresponding address width of the AXI master and slave ports. One address rule per cache way gets defined. This is one of the reasons why the LLC parameters NoLines and NoBlocks have to be a power of two. It then provides a continuous address mapping for the SPM region. The SPM addressing follows the normal AXI byte addressing scheme, as a SPM access uses the same pipeline as a cached accesses.

When a transaction enters the slave port of the LLC, the address mapping for the RAM, the address mapping for the SPM region and the current SPM configuration determine the selection signal which controls the following axi_demux module. The module provides the functionality that transactions can bypass the cache if all of its ways are configured as SPM. The mapping also ensures that accesses onto the SPM region always go into the pipeline. When an AXI AW or AR beat get send towards the cache pipeline it enters the respective axi_llc_chan_splitter module. This module has the responsibility to generate all following LLC descriptors, mapping the whole AXI transfer onto the respective cache-lines affected.

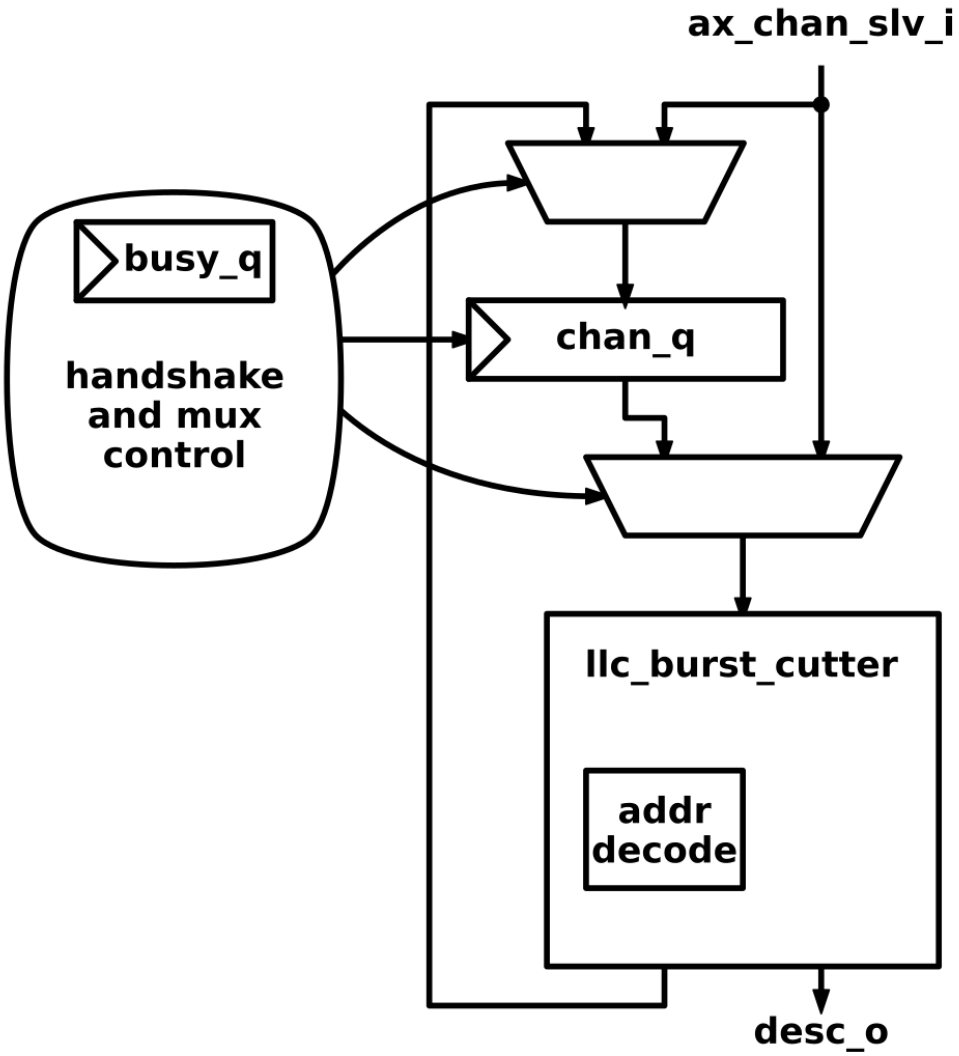

Figure 4. Block-diagram of the channel splitter.

The first descriptor gets sent towards the hit/miss spill register in a combinational way. This is to reduce the latency of the design. The spill register before the hit/miss detection is purely added to cut the longest logic path. The splitter unit then shaves off descriptors from the control AXI vector, until all required cache descriptors are sent. This is done by the submodule axi_llc_burst_cutter. This combinational block defines the address of the descriptor, how many data beats of the AXI transaction go onto this particular cache-line and what is left for all subsequent descriptors. The module basically defines sub-bursts which neatly fit exactly onto one cache line. From the splitter the descriptors for read and write transactions get merged together with the data path of the flush descriptors. This merging of the accesses has to be done as the tag lookup can only handle one descriptor at a time and to ensure that the tag state is unified for both read and write

transactions. This merging has only marginal impact on the throughput of long bursts. The reason being that the LLC normally is the last cache before the main memory. This implies that most of its accesses are bursted. For example when a cache more closely to the CPU gets its cache line replaced. This means that the total number of AXI control vectors from the CPU side will be lower than the total amount of data beats associated with these vectors. Merging now the control vectors leads to a better utilization of the following logic in the pipeline. Another responsibility of the axi_llc_burst_cutter module is to determine the final mapping of the SPM region. With the way_ind field of the descriptor the read, write and miss pipeline modules determine the exact data-way towards the respective accesses should be routed.

6.4.2.0.2 Hit and Miss Detection

Coming from the splitter modules and the spill register the descriptor enters the axi_llc_hit_miss detection unit. This unit is responsible for the tag lookup and determines if a descriptor is allowed through the hit bypass or not. It has three sub components. First the axi_llc_tag_store module, which does the lookup of the tag. The axi_llc_lock_box_bloom module which ensures that only one descriptor is allowed to operate on the same cache line at a time. And lastly the axi_llc_miss_counters module, which is responsible of the right ordering of the descriptors. The sub-modules are handshaked to enable back pressure.

The module gets initialized each time when the LLC comes out from a reset. The tag-storage macros have to be initialized to ensure correct behavior. During this initialization phase the tag-storage module runs a build in self test (BIST) on the tag-storage SRAM macros. The BIST is run in parallel on all macros and reuses the lookup comparators, which have to be there anyway to make the tag lookup work. The BIST uses the *march X* pattern to check for eventual defects. The advantage of using this pattern is that in the end all macros are initialized to zero and no further initialization steps are required. The result of the BIST can then be read out in the respective configuration address over the AXI LITE port.

6.4.2.0.2.1 Tag Lookup and Storage

The axi_llctag_store module houses the SRAM macros which are responsible for the tag management. It is fully handshaked, providing back pressure capabilities. It has multiple request modes.

- axi_llc_pkg::BIST: All requests during initialization have this type. All macros get accessed and the contents compared in the next cycle.
- axi_llc_pkg::LOOKUP: Performs a lookup on the SRAM macros which are not configured as SPM. The unit then responds either with a hit, or a miss. On a hit the way indicator is given back. On a miss the module decides with help of the evict_box, which way should be used for storing the line which has to be refilled.
- axi_llc_pkg::STORE: Performs a storage of the tag onto the SRAM macro indicated by the input way indicator. It also sets the dirty flag if it was issued from a write descriptor.
- axi_llc_pkg::FLUSH: Performs a lookup on only one of the macros, and determines if this cache line has to be evicted, when its dirty flag is set. Then the tag gets overwritten by all zeros.

A lookup has one cycle latency as the SRAM macros have to be pipelined for getting acceptable cycle timings. In parallel to the macros there is a register which stores the tag for which the lookup is performed. After lookup a comparator for each set compares the tag returned from the macro with the one stored in the flipflop for determining a hit. These comparators also perform the BIST at initialization of the storage. On a miss, the valid flags get checked against the SPM configuration to determine if there is space in one of the sets for the to be refilled cache-line. Currently a small shift register as long as the set-associativity of the cache is responsible of finding the next way where the line should go. This register is advancing every clock cycle, providing a pseudo random chosen way. When there are still empty positions, the shift-register will stop at the next empty position and will wait for the output handshake to occur. If there are no empty positions left it stops at the next non SPM configured position. When there is a cache line to be replaced, the dirty flag is also returned, defining if the eviction flag in the descriptor has to be set.

6.4.2.0.2.2 Handling of Hits and Misses

After the tag lookup all actions which have to be performed on a specific cache line are known. Its exact location, if it as to be refilled, etc. There has to be made sure that different accesses do not interfere with each other. One of the least favorable examples would be that one AXI transaction from the slave port is currently writing onto a cache line, and a subsequent reading transaction into another memory region evicts this currently in use cache-line. Then the written data could overwrite the new one, leading to memory corruption. Another concern is the preservation of the AXI ordering model, as no descriptor with the same ID is allowed to overtake another. This could happen if an AXI burst accesses multiple cache-lines, the first one misses, the second one now taking the bypass, leading to a not allowed reordering.

For first problem of accidental eviction, a cache line locking mechanism is introduced. Its function is to ensure that only one descriptor is allowed to work on a specific cache-line at time. When a descriptor leaves the hit/miss detection unit the line is locked. When the descriptor leaves the pipeline by completing its operations in the read or write unit the line gets unlocked. When a descriptor in the hit/miss unit wants to access a locked cache line, it gets stalled until the cache line is unlocked again. For the locking a counting bloom-filter is used. This data structure has the advantage that it does not change in size when the cache size increases and the lookup does not need many logic levels. Its size depends on the desired false positive rate, which itself in this case depends on the amount of currently locked cache lines. This has a maximum value as it solely depends on the number of descriptors which can be downstream from the detection unit. It is determined by the amount of units and their respective descriptor FIFO's depth which can be adapted in axi_llc_pkg. The key thing for using a counting bloom filter here is its property to not have false positives, ensuring that no descriptor can advance, when the line it tries to access is locked. False positives only lead to a decrease in throughput of the descriptors. It can be observed during simulation that false positives lead to no significant drop in data throughput. The reason is that when a false positive occurs, other descriptors have to be downstream as they are the cause for false positive in the first place. It then leads to the bloom filter gradually emptying, until the colliding descriptor unlocks its cache-line allowing the descriptor to advance.

The ordering is the second issue which has to be solved. The hit bypass is designed to allow descriptors to overtake each other. However the ordering properties of the AXI protocol have to be upheld. For ensuring proper ID ordering a number of ID counters are introduced. They operate by taking in account the lower bits of the ID. This property can be changed in the axi_llc_pkg. The counters track, how many descriptors of which ID are currently in the miss pipeline operating. There is a separate counter which tracks all write descriptors as they have to be strictly transmitted in order. When a descriptor hits, the counters get checked for its ID. If there are other descriptors with the same ID in currently in the miss pipeline, the descriptor is not allowed through the bypass and instead passed towards it. As none of its eviction or refill flags will be set, no action is performed in the pipeline, passing through the descriptor. When a descriptor leaves the pipeline its respective counter gets decreased.

6.4.2.0.2.3 Data Read and Write

Each descriptor generated from a transaction on the slave port ultimately ends up in either the read or the write unit. They control the other tree channels on the slave port. Their responsibilities is to make the requests on the data-ways. On descriptor load, internal counters get set with the amount of requests, which have to be made to the data-ways. The request towards them are bundled in the struct outlined in following table:

Name	Type	Function
cache_unit	llc_pkg::unit_e	Indicated which of the units made the request towards the data-ways. This value is used for routing back the response from an individual way.
way_ind	logic [SetAsso]	Indicates to which way the request should go. Controls the request routing inside llc_data_ways.
line_addr	logic [Index]	The cache line index indicating which cache line the request is for.
block_offset	logic [BlockOffs]	The block offset indication to which cache block the request is for.
we	logic [SetAsso]	Is the request a write [1'b1] request or a read [1'b0] request.
data	logic [DataWidth]	On write contains the data to be written to a block. On read this signal is ignored by the way.
strb	logic [DataWidth/8]	The write strobe signal associated which bytes of the block are valid for writing.

There are four units in the LLC which can make requests to the data-ways:axi_llc_read_unit, axi_llc_write_unit, axi_llc_refill_unit and axi_llc_evict_unit. The units which make read requests get their responses back from the data-ways. It further features the needed request interconnect which makes it possible that each of the four units can make a request to a macro, if the request go towards different macros. This allows the cache to saturate each of the data channels in both read and write direction on the slave and master port.

When a request enters the LLC-ways, it gets routed towards the way indicated by its way-indicator. For the unit issuing read requests there exists a FIFO, which stores the way_ind of the request. This FIFO is necessary as read requests have the possibility of going from the same unit to two different ways, when the descriptor is changing. These FIFOs control the read response switching back to the modules and ensure proper ordering. Each data-way is handshaked in both request and response direction. They are designed to handle one request per cycle. When there is back pressure from the response side, subsequent request will also be stalled. For each unit exists a separate multiplexer and can forward the responses to the units in parallel.

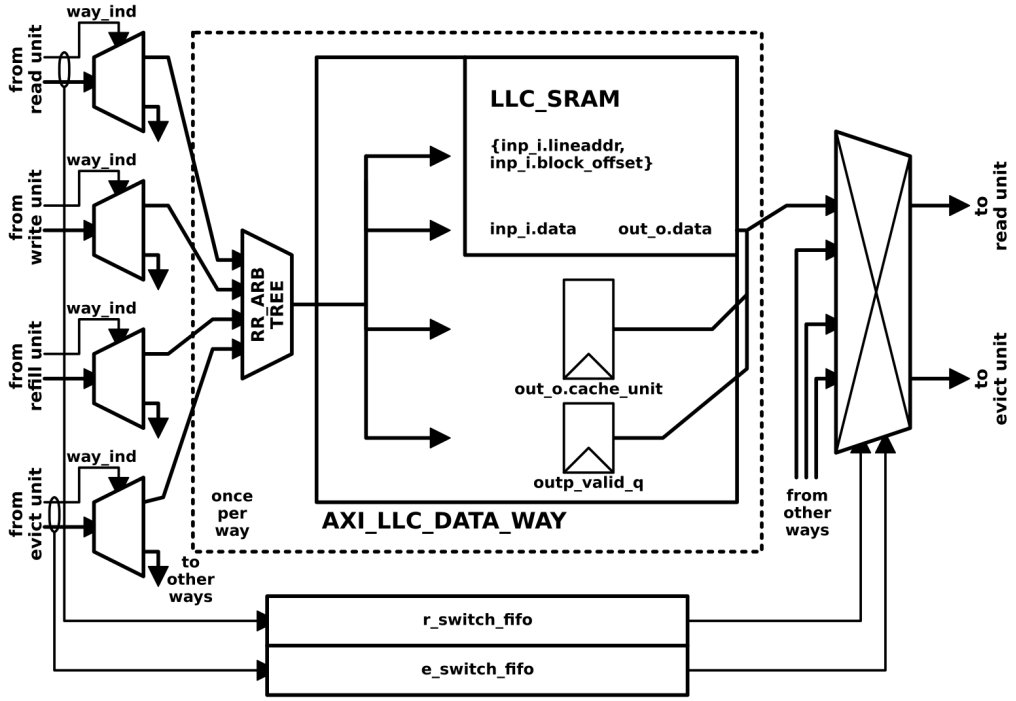

Figure 5. Block-diagram of the basic setup of the data storage.

6.4.2.0.3 Eviction and Refilling

On a miss or to preserve ordering of an AXI burst, the descriptor goes from the hit miss detection unit onto the so called miss pipeline. The pipeline has two functions. The first is evicting a cache-line and the second is refilling it. A descriptor passing through gets checked for the evict or refill flag. When a flag is set the unit will do their respective operations. When no flag is set the descriptor only gets passed along.

Entering the pipeline the descriptor goes to the eviction unit. The unit has three subcomponents in it and is responsible of writing a whole cache-line towards the main memory using the master port of the LLC. The first submodule initiates an AW vector on the port if the eviction flag is set on the descriptor. The burst type is always INCR and the burst length is always the number of blocks inside a line, as the job of this unit is to write back the entirety of a cache line. The ID is also always the same, as the writing operation on the W channel anyways has to be done in order. It also prevents reordering of the responses on the master ports B channel. The descriptor then gets sent further into a FIFO which is there to buffer multiple requests. Then the descriptor enters the unit which is responsible of reading out the data to be evicted. The unit takes the address of the descriptor and the way-indicator and calculates the read requests which have to be made to the data-ways to read out the entire cache-line. The responses from the data-way then get sent back to the unit and first enter a FIFO, which is connected to the data field of the W channel. This FIFO ultimately also controls the handshaking of the W channel. The unit has a counter, which determines, when the unit has to send the last flag. The unit then waits for the B response and sends the descriptor further. When the descriptor has its flush flag set, instead then the descriptor gets destroyed and the control unit notified that a line was flushed.

After the evict unit the descriptor enters the refill unit. It has the reverse function of the eviction unit as it is responsible foe cache-line refills. The descriptor enters the unit which controls the AR channel and makes a read request for the whole cache-line, when the refill flag is set. The unit initiates the AXI transaction on the AR channel, again always with the same ID to prevent reordering from the read responses. The descriptor the enters a FIFO for buffering multiple descriptors to enable the cache to have multiple refill requests in flight at the same time. When the descriptor the enters a submodule which sits on the R channel of the master port. On descriptor load, internal counters get set for controlling the write requests towards the data-ways. Here no FIFO is present, the unit however connects the handshaking from the master ports R channel to the one towards the data-ways. When all data beats are transmitted the descriptor gets sent then either to the read or the write unit. When this transfer occurs the merge unit signals towards the miss counters in the hit miss detection unit that a descriptor with its respective ID has left the pipeline. There then the miss count gets decreased. On reaching zero, hits are allowed again for using the hit bypass.

6.4.2.0.4 Configuration and Flush

The axi_llc_config module is the unit which controls the SPM configuration and the flushing of the LLC. It also feature a set of performance counters, which can be used to track the descriptor utilization of the LLC. These functionalities all can be accessed over a dedicated AXI4 LITE slave port.

There are three registers which can be written to and are located at the lower part of the address map. The map gets generated automatically from a combination of lite port data width and the start address of the configuration map. These registers are also responsible for the maximum set-associativity of the cache. As the SPM configuration is encoded as a one-hot signal each way index matches to a bit in the respective register. As the register is a wide as the LITE port data width the maximum set-associativity is in consequence also limited to that value. This mapping makes the read out of the BIST from the tag storage macros trivial. The set/way mapping is the same for SPM

configuration, BIST readout and manual flush initialization, where each bit index matches the way index.

Flushing of the cache can also be initiated over this LITE configuration port. When a bit in the flush register is set, a finite state machine (FSM) gets activated. When changing the SPM configuration this state machine further gets activated automatically, writing back all affected dirty data to the memory.

The FSM is not optimized for speed, as flushing usually is not done often. Instead it is designed to ensure maximum safety of the flush operation so that there is no chance data will be corrupted. When the FSM gets activated, no new write transactions can be sent to the LITE configuration port. It is however possible to read out the values. For this, on activation the FSM goes from IDLE to one of three states, depending on the current state of the slave ports Ax channels. It waits in these states until all initiated requests on the slave port are transmitted, and setting the ready signal to zero, so that no new requests can be made to the cache during flushing. The FSM then waits until no new descriptors get generated in the splitter units. When they are empty the flushing gets initialized by determining which of the ways need flushing.

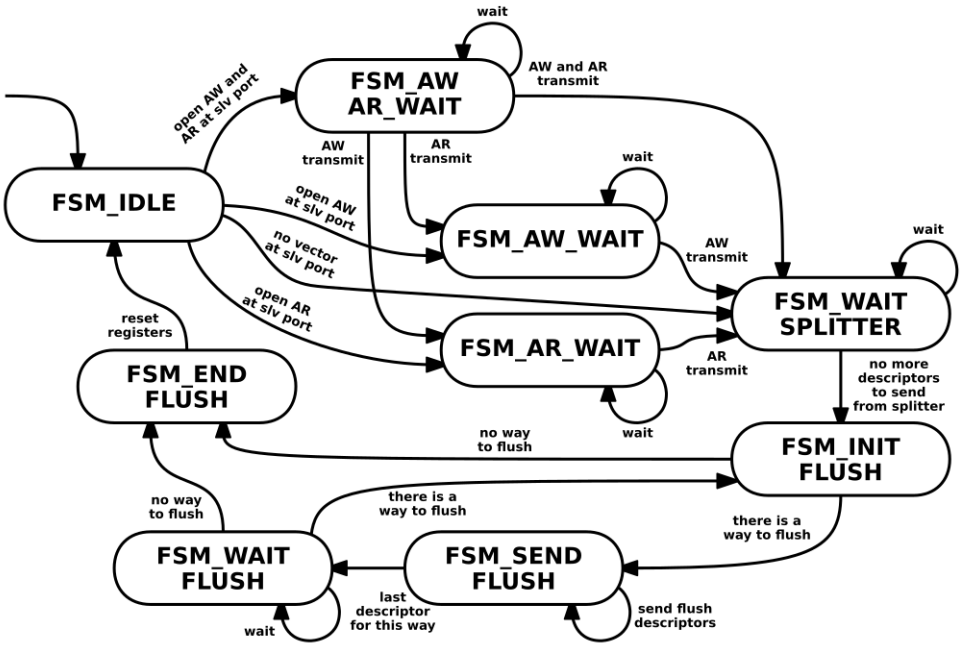

Figure 6. Diagram of the state machine controlling the flush operation.

6.4.2.1 LLC Configuration Registers Summary

Config cache address range start (LLC_CACHE_ADDR_START)
0x1A106018

Config Cache address range end (LLC_CACHE_ADDR_END)
0x1A10601C

Config SPM address range start (LLC_SPM_ADDR_START)
0x1A106020

Config cache way mode: SPM/cache (CFG_SPM_LOW)
0x10401000

Config cache way flush (CFG_FLUSH_LOW)
0x10401008

Config cache way flush status (FLUSHED_LOW)
0x104010010

6.4.2.2 Default LLC Configuration

The following is an example of how SW could set the default LLC configuration.

Default cached region address start:
0x1A106018 0x80000000

Default cached region address end:

0x1A10601C 0x80800000

Default SPM region address start:
0x1A106020 0x70000000

Entire LLC set in cache mode:
0x10401000 0x0

6.4.2.3 LLC Bypassing

To bypass the LLC and access the HyperRAM directly, you can set the entire LLC (all 8 ways) in SPM mode. To do so, you need to write 0xFF to the CFG_SPM_LOW register (address 0x10401000): 0x10401000 0xFF

6.5 PULP Cluster

The cluster domain features the Parallel Ultra-Low Power (PULP) cluster and the RedMule Tensor Unit integrated as an additional Hardware Processing Engine (HWPE).

- RISC-V cores - 8 x CV32E40P cores
- Tensor Unit - RedMulE
- DMA
- SRAM banks - 256KB
- Instruction Cache – L1 (512B) and L1.5 (4kB)
- HW synchronizer
- AXI Bus with dual-clock FIFOs for connection to the SoC
- Hybrid interconnect

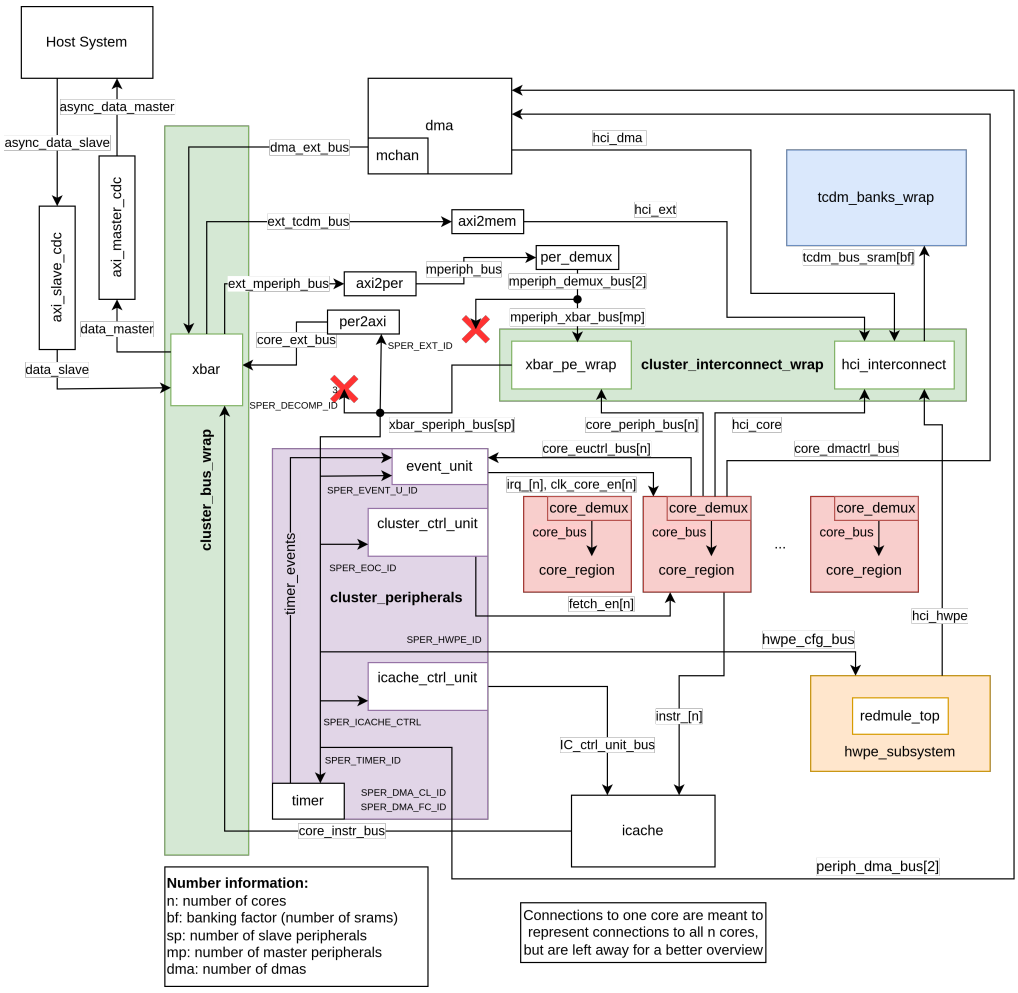

Figure 7. AlSaqr Cluster

The PULP cluster comprises 8 4-stage in-order 32b RISC-V RI5CY cores, whose ISA was extended to support multiple additional instructions including hardware loops, post-increment load and store instructions and additional ALU instructions. Moreover, each core features a dedicated Floating Point Unit (FPU) supportingthe FP32, FP16 and FP16ALT formats.

The instruction interface is connected to a hierarchical instruction cache consisting of one private bank per core, each configured to store 512 B. Each private bank fetches instructions from a larger, shared cache of 4 KiB, improving the performance of applications using the Single Program Multiple Data (SPMD) paradigm.

In addition, each core features a data interface that connects it to the rest of the system for direct access to the system's Tightly-Coupled Data Memory (TCDM) and all other memory-mapped peripherals. The TCDM comprises 256KB organised in 16 32 bit word-interleaved memory banks providing single-cycle access latency through the Heterogeneous Cluster Interconnect (HCI), which features a logarithmic branch that allows all-to-all accesses from 32-bit master ports, like those of the cores or the DMAC, to each of the word-interleaved memory banks.

A banking factor of 2 (i.e., the number of memory banks is twice the number of cores) is used to minimize the memory banks contention probability even for highly memory-intensive workloads. If a collision occurs, round-robin arbitration guarantees fairness and avoids starvation.

Along with the connection to the TCDM, the cores have access to both memory-mapped devices within the cluster and the host domain through a peripheral interconnect.

One of the core-local peripherals is the Direct Memory Access (DMA) unit, capable of up to 64 bit/cycle data transfers in either direction, full duplex between the external larger memories in the host subsystem and the cluster's local TCDM.

Finally, the cores directly connect to an event unit responsible for synchronization barriers within the cluster. Each core attempts to read from the corresponding register within the event unit and only receives a response once all cores request the same barrier address. Furthermore, the event unit manages interrupts within the cluster, masking and forwarding incoming interrupt signals to the responsible core.

To ensure the cluster can easily access the host system, the cluster has both an input and an output AXI port connected to an AXI interconnect. In particular, the communication is asynchronous and it leverages a pair of input and output Cross Domain Clock (CDC) FIFOs. Through this interconnect, the host can access the cluster's internal memory and peripherals for configuration. Viceversa, the instruction cache, the DMA, and the cores have access to the host system's memory.

6.5.1 RedMulE Tensor Unit

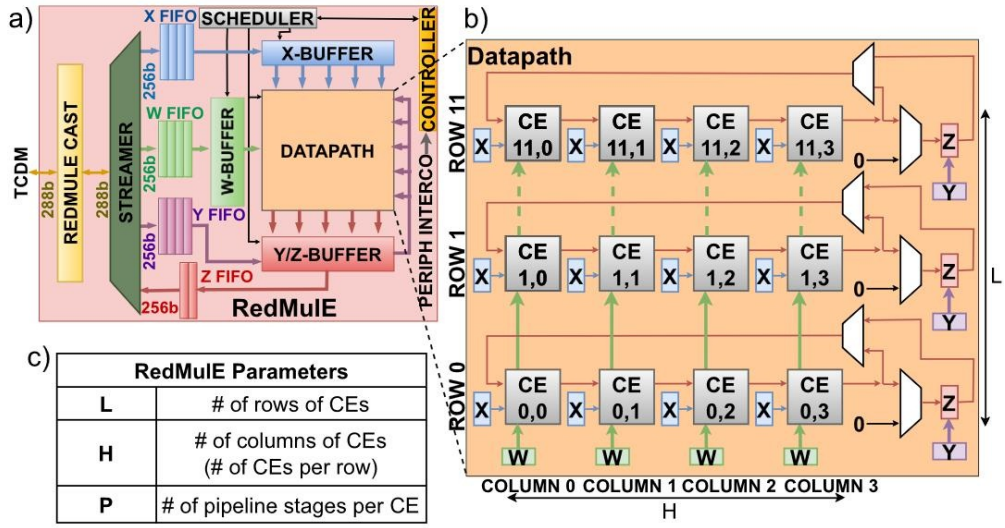

Figure 8. RedMulE Tensor Unit

RedMulE (Reduced-Precision Matrix Multiplication Engine) is a low-power specialized accelerator conceived for multi-precision floating-point General Matrix–Matrix Operations (GEMM-Ops) acceleration, supporting FP16, as well as hybrid FP8 formats, with {sign, exponent, mantissa} = ({1, 4, 3}, {1, 5, 2}). RedMulE allows the instantiation of a wide range of Floating-Point Units-based Computing Elements (CEs), internal buffers, and memory interface configurations.

The core of RedMulE is the Datapath, a 2-Dimensional array of CEs interconnected. The CEs are organized in L rows, each made of H columns. Within each row, a number of H CEs are cascaded so that each CE computing an intermediate product will pass its result to the next CE. The partial product computed by each row's last CE is fed back as accumulation input of the same row's first CE. The RedMulE Datapath features a design-time configurable number of internal CEs, pipeline registers (P) for each CE, and internal computing precision (FP bitwidth).

It has been integrated in the cluster as an additional HWPE to further extended the cluster capabilites choosing H = 4, L = 3 and P = 12.

It is software programmed by the RISC-V cores through the peripheral interconnect and it shares the TCDM with the RISC-V cores and the DMAC, making it tightly-coupled with the cluster cores. This kind of integration uses a shallow branch in the HCI interconnect which features a single port, routed to adjacent 32-bit memory banks treated like a single wider bank without arbitration. The TCDM banks are connected to the logarithmic and shallow HCI branches through a set of multiplexers, which grant access to one branch or the other according to a configurable starvation-free rotation scheme, allocating a configurable maximum of K < N consecutive cycles to the HWPE over a period of N cycles.

6.5.2 Event Units

There is 1 event unit (EU) available in Al Saqr'scluster.

The EU allows the CV32E40P cores to be put into sleep mode when waiting for an event to occur. In the EUs, the way of treating incoming events can be controlled. The EU can be instructed to react instantly by jumping to an interrupt routine or to delegate the treatment of the event to a software event task controller.

6.6 DMA (direct memory access)

The DMA unit allows the transfer of data between L2 and cluster L1 memory areas. 8 channels can be programmed. Channels can be 1D/2D on the L2 memory and 1D on the cluster L1 side.

6.7 Debug architecture

To debug the Al Saqr chip, a JTAG and UART are provided. The L2SPM memory region is an executable and cacheable region for CVA6. Such a setup provides the minimal memory space to have the chip working standalone, avoiding to rely on the off-chip HyperRAMs for initial testing.

Al Saqr contains debug functionalities to help the developer observing/controlling application code execution. Debug functionalities are accessible through JTAG or SPI Slave interfaces using a GDB server.

Further documentation regarding the RISC-V debug module can be found atthis page

6.8 Micro DMA

The micro DMA (uDMA) provides direct transfer of data between the different interfaces available in Al Saqr, and connected to the uDMA, and the L2 memory and the L3 off-chip memory. Furthermore, the uDMA supports data transfers between the L2 and the L3. The latter communicates through a HyperBus interface. Up to 11 channels can be managed by the uDMA:

- 11 x SPI from/to L2 and L3
- 1 x QSPI from/to L2 and L3
- 3 x UART from/to L2 and L3
- 4 x USART from/to L2 and L3
- 2 x SDIO from/to L2 and L3
- 6 x I2C from/to L2 and L3
- 2 x CPI to L2 and L3
- 1 x HYPERBUS from/to L2 (to/from L3)

The width of transfers can be selected between 8, 16 or 32 bits. Up to 128kB can be transferred during a single transaction (8kB for SPIM). In the general case, transactions can be bidirectional but depending on the interface, in some cases only one direction is available.

6.8.1 SPI master (serial peripheral interface)

Up to 11 SPI master interfaces are available:

- The SPI (Master) is able to communicate at speeds up to 50Mbits/s. It uses four lines: MISO (Master In, Slave Out), MOSI (Master

Out, Slave In), SCK (Serial Clock), and SS (Slave Select). It is a full-duplex communication protocol, allowing for simultaneous data transmission and reception.

6.8.2 QSPI master (quad serial peripheral interface)

1 QSPI master interface is available:

- The QSPI (Master) can be used as Single or Quad SPI. it is able to communicate at speeds up to 50Mbits/s. When used as Singles, refere to the documentation for the SPI. When used as Quad it enhances SPI by using up to six lines: four for data (IO0, IO1, IO2, IO3), one for clock (SCK), and one for slave select (SS). QSPI can operate in a 4-bit wide quad mode, allowing it to transmit four bits per clock cycle. Under the hood the SPI and QSPI are the same hardware IP. The difference between the two is that the SPI exposes only two data lines (MISO and MOSI) of the available four (IO0, IO1, IO2 and IO3).

6.8.3 UART (universal asynchronous receiver-transmitter)

Up to 3 UART interfaces are provided with up to 1Mbits/s baud rate. No dedicated synchronization (CTS/RTS/DTR/DSR/DCD) signals are provided.

6.8.4 USART (universal synchronous-asynchronous receiver-transmitter)

Up to 4 USART interfaces are provided with up to 1Mbits/s baud rate.

6.8.5 SDIO (Secure Digital Input Output)

Up to 2 SDIO (Secure Digital Input Output) are provided in Al Saqr.

6.8.6 I2C (inter-integrated circuit)

Up to 6 I²C (Inter-Integrated Circuit) are provided in Al Saqr. They support multi-master, multi-slave, single-ended modes. I²C uses only two bidirectional open-drain lines, Serial Data Line (SDA) and Serial Clock Line (SCL), pulled up with resistors.

6.8.7 CPI (camera parallel interface)

Up to 2 CPI interface are 8 bits wide and can communicate at speeds up to 50MHz. VSYNC, HSYNC and PCLK are provided by the camera.

6.8.8 HyperBus

1 HyperBus interfaces running at 200MHz, up to a maximum of 6.4Gbps. The HyperBus interface is a high-speed, low-latency interface designed for interfacing with memory devices. It supports the fully digital HyperBus protocol used by the HyperRAM controller. The interface features eight bi-directional Double-Data-Rate (DDR) data lines (DQ[0:7]), an active-low chip select (CSn) for each memory bank, a read-write data strobe (RWDS), the clock line and its inverse (CK and CKn) and an active-low reset line (RESETn). Further details can be found here

6.9 GPIOs (general purpose inputs/outputs)

Up to 32 digital general purpose I/Os are available. Each I/O can be configured either as an input or output. Interrupts on event can be generated on the rising or the falling or both edges for all I/Os. I/Os can also be configured to act as an external wake up signal.

6.10 CAN (Controller Area Network)

Up to 2 CAN bus which is a robust vehicle bus standard designed to allow microcontrollers and devices to communicate with each other's applications without a host computer. It is a message-based protocol, designed originally for multiplex electrical wiring within automobiles to save on copper, but it can also be used in many other contexts. For each device, the data in a frame is transmitted sequentially but in such a way that if more than one device transmits at the same time, the highest priority device can continue while the others back off. Frames are received by all devices, including by the transmitting device. Further details can be found at at this page

6.11 Timers

There are 2 basic timers are available, one connected to the CVA6 cores and the other to the Cluster. They can be configured either as 2 x 32-bit timers or as a single 64-bit timer. The basic timers can either run continuously or trigger just once. Events can be generated using a compare match.

Clock sources of these timers can be the:

- FLL
- FLL with pre-scaler
- 32.768kHz reference clock

6.12 PWM Timers

4 advanced PWM timers are available in the SOC domain. Each of them provides 4 output signal channels that can be used for PWM signal generation with multiple configuration possibilities.

6.13 RTC

A real-time clock is available. It provides a set of continuously running counters which can be used with suitable software to provide a clock calendar function. It provides also alarm and a periodic interrupt features. It is clocked by the 32.768 kHz external crystal.

6.14 Performance counters

Each CV32E40P cores of the cluster provide a performance counter. These 32-bit counters can be configured to count the:

- Total number of cycles (also includes the cycles where the core is sleeping)
- Number of cycles the core was active (not sleeping)
- Number of instructions executed
- Number of load data hazards
- Number of jump register data hazards
- Number of cycles waiting for instruction fetches, i.e. number of instructions wasted due to non-ideal caching
- Number of data memory loads executed. Misaligned accesses are counted twice
- Number of data memory stores executed. Misaligned accesses are counted twice
- Number of unconditional jumps (j, jal, jr, jalr)
- Number of both taken and not taken branches
- Number of taken branches
- Number of compressed instructions executed
- Number of memory loads to EXT executed. Misaligned accesses are counted twice. Every non-L1 access is considered external (cluster only)
- Number of memory stores to EXT executed. Misaligned accesses are counted twice. Every non-L1 access is considered external (cluster only)
- Number of cycles used for memory loads to EXT. Every non-L1 access is considered external (cluster only)
- Number of cycles used for memory stores to EXT. Every non-L1 access is considered external (cluster only)
- Number of cycles wasted due to L1/log-interconnect contention (cluster only)
- Number of cycles wasted due to CSR access

7 Memory map

The following table describes Al Saqr's memory map. All areas in this map are addressable from any cores.

			Address range
Debug Unit			0x00000000 - 0x00000FFF
	Debug Unit		0x00000000 - 0x00000FFF
ROM Memory			0x00010000 - 0x0001FFFF
	ROM (64kB)		0x00010000 - 0x0010FFFF
Host Subsystem			0x02000000 - 0x0FFFFFFF
	Core Local Interrupt (CLINT)		0x02000000 - 0x020BFFFF
	Platform Level Interrupt Controller (PLIC)		0x0C000000 - 0x0FFFFFFE
Cluster Subsystem			0x10000000 - 0x1040FFFF
	Cluster L1 RAM (256kB)		0x10000000 - 0x1003FFFF
	Cluster L1 memory test and set unit		0x10100000 - 0x101FFFFF
	Cluster Control Unit		0x10200000 - 0x102003FF
	Cluster timer		0x10200400 - 0x102007FF
	Cluster event unit		0x10200800 - 0x10200FFF
	Hardware Processing Element		0x10201000 - 0x102013FF
	Cluster icache control		0x10201400 - 0x102017FF
	DMA		0x10201C00 - 0x10201FFF
	Cluster Demux Event Unit		0x10204000 - 0x102043FF
	Cluster Demux MCHAN		0x10204400 - 0x102047FF
Axi Lite Subsystem			0x10400000 - 0x1040FFFE
	Cluster TLB Config Registers		0x10400000 - 0x10400FFF
	Last Level Cache Config Registers		0x10401000 - 0x10401FFF
	Host to PULP Mailbox		0x10402000 - 0x10402FFF
	Cluster to PULP Mailbox		0x10403000 - 0x10403FFF
	OpenTitan Mailbox		0x10404000 - 0x10404FFF
System Timer			0x18000000 - 0x18000FFF
	System Timer		0x18000000 - 0x18000FFF
APB Subsystem			0x1A100000 - 0x1A231000
	SoC FLL		0x1A100000 - 0x1A100FFF
	HYPERBUS_AXI		0x1A101000 - 0x1A101FFF
	PWM0		0x1A103000 - 0x1A103FFF
	PWM1		0x1A223000 - 0x1A223FFF
	PWM2		0x1A224000 - 0x1A224FFF
	PWM3		0x1A225000 - 0x1A225FFF
	PWM4		0x1A226000 - 0x1A226FFF
	PWM5		0x1A227000 - 0x1A227FFF
	PWM6		0x1A228000 - 0x1A228FFF
	PWM7		0x1A229000 - 0x1A229FFF
	Padframe Config		0x1A104000 - 0x1A104FFF
	GPIO Config Registers		0x1A105000 - 0x1A105FFF
	SoC Control Registers		0x1A106000 - 0x1A106FFF
	CAN 0		0x1A108000 - 0x1A108FFF
	CAN 1		0x1A109000 - 0x1A109FFF
	MicroDMA Subsystem		0x1A200000 - 0x1A21EFFF
		UDMA control	0x1A200000 - 0x1A200FFF
		UART Channel 0	0x1A201000 - 0x1A201FFF
		UART Channel 1	0x1A202000 - 0x1A202FFF
		UART Channel 2	0x1A203000 - 0x1A203FFF
		USART Channel 0	0x1A204000 - 0x1A204FFF
		USART Channel 1	0x1A205000 - 0x1A205FFF

			Address range
		USART Channel 2	0x1A206000 - 0x1A206FFF
		USART Channel 3	0x1A207000 - 0x1A207FFF
		SPI Master Channel 0	0x1A208000 - 0x1A208FFF
		SPI Master Channel 1	0x1A209000 - 0x1A209FFF
		SPI Master Channel 2	0x1A20A000 - 0x1A20AFFF
		SPI Master Channel 3	0x1A20B000 - 0x1A20BFFF
		SPI Master Channel 4	0x1A20C000 - 0x1A20CFFF
		SPI Master Channel 5	0x1A20D000 - 0x1A20DFFF
		SPI Master Channel 6	0x1A20E000 - 0x1A20EFFF
		SPI Master Channel 7	0x1A20F000 - 0x1A20FFFF
		SPI Master Channel 8	0x1A210000 - 0x1A210FFF
		SPI Master Channel 9	0x1A211000 - 0x1A211FFF
		SPI Master Channel 10	0x1A212000 - 0x1A212FFF
		QSPI Master Channel 0	0x1A213000 - 0x1A213FFF
		I2C Channel 0	0x1A214000 - 0x1A214FFF
		I2C Channel 1	0x1A215000 - 0x1A215FFF
		I2C Channel 2	0x1A216000 - 0x1A216FFF
		I2C Channel 3	0x1A217000 - 0x1A217FFF
		I2C Channel 4	0x1A218000 - 0x1A218FFF
		I2C Channel 5	0x1A219000 - 0x1A219FFF
		SDIO Channel 0	0x1A21A000 - 0x1A21AFFF
		SDIO Channel 1	0x1A21B000 - 0x1A21BFFF
		CAM channel 0	0x1A21C000 - 0x1A21CFFF
		CAM channel 1	0x1A21D000 - 0x1A21DFFF
		Filter	0x1A21E000 - 0x1A21EFFF
		HYPERBUS Reister file 0	0x1A21F000 - 0x1A21FFFF
		HYPERBUS Reister file 1	0x1A220000 - 0x1A220FFF
L2 Memory			0x1C000000 - 0x1C008000
	L2 RAM (32kB)		0x1C000000 - 0x1C007FFF
SoC Peripherals Subsystem			0x20000000 - 0x40000000
	QSPI Linux		0x20000000 - 0x207FFFFF
	Ethernet		0x30000000 - 0x300FFFFF
	Core UART		0x40000000 - 0x40000FFF
Last Level Cache SPM			0x70000000 - 0x70020000
	Last Level Cache		0x70000000 - 0x7001FFFF
L3 Memory			0x80000000 - 0xA0000000
	L3 Memory		0x80000000 - 0x9FFFFFFF

Table 4. Pulp memory map table

8 Device components description

8.1 Cluster Subsystem

8.1.1 Cluster control unit

None

8.1.1.1 Cluster Control Unit registers

Name	Address	Size	Type	Access	Default	Description
EOC	0x10200000	32	Status	R/W	*0x0000*	End Of Computation status register.
FETCH_EN	0x10200008	32	Config	R/W	*0x0000*	Cluster cores fetch enable configuration register.
MISC_CTRL	0x10200018	32	Config	R/W	*0x0000*	Miscellaneous Control
CLOCK_GATE	0x10200020	32	Config	R/W	*0x0000*	Cluster clock gate configuration register.
DBG_RESUME	0x10200028	32	Config	W	*0x0000*	Cluster cores debug resume register.
DBG_HALT_STATUS	0x10200028	32	Config	R	*0x0000*	Cluster cores debug halt status register.
DBG_HALT_MASK	0x10200038	32	Config	R/W	*0x0000*	Cluster cores debug halt mask configuration register.
BOOT_ADDR0	0x10200040	32	Config	R/W	*0x0000*	Cluster core 0 boot address configuration register.
TCDM_ARB_POLICY_CH0	0x10200080	32	Config	R/W	*0x0000*	TCDM arbitration policy ch0 for cluster cores configuration register.
TCDM_ARB_POLICY_CH1	0x10200088	32	Config	R/W	*0x0000*	TCDM arbitration policy ch1 for DMA/HWCE configuration register.
TCDM_ARB_POLICY_CH0_REP	0x102000C0	32	Config	R/W	*0x0000*	Read only duplicate of TCDM_ARB_POLICY_CH0 register
TCDM_ARB_POLICY_CH1_REP	0x102000C8	32	Config	R/W	*0x0000*	Read only duplicate of TCDM_ARB_POLICY_CH1 register

Table 5. Cluster Control Unit registers table

8.1.1.2 Cluster control unit registers details

8.1.1.2.1 End Of Computation status register. (EOC)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															EOC

Bit 0 - **EOC** *(R/W)*

End of computation status flag bitfield:

- *0b0*: program execution under going
- *0b1*: end of computation reached

8.1.1.2.2 Cluster cores fetch enable configuration register. (FETCH_EN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CORE7	CORE6	CORE5	CORE4	CORE3	CORE2	CORE1	CORE0

Bit 7 - **CORE7** *(R/W)*

Core 7 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 6 - **CORE6** *(R/W)*

Core 6 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 5 - **CORE5** *(R/W)*

Core 5 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 4 - **CORE4** *(R/W)*

Core 4 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 3 - **CORE3** *(R/W)*

Core 3 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 2 - **CORE2** *(R/W)*

Core 2 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 1 - **CORE1** *(R/W)*

Core 1 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 0 - **CORE0** *(R/W)*

Core 0 fetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.1.2.3 Miscellaneous Control (MISC_CTRL)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			FREGFILE_DIS	HWPE_EN	HCI_LOW_PRIO_MAX_STALL		HCI_HWPE_PRIO	Reserved							

Bit 12 - **FREGFILE_DIS** *(R/W)*

Disable Floating Point Register File bitfield:

- *0b0*: enabled
- *0b1*: disabled

Bit 11 - **HWPE_EN** *(R/W)*

Enable HWPE bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bits 10:9 - **HCI_LOW_PRIO_MAX_STALL** *(R/W)*

Maximum numbers of stalls before the HWPE request is served in the interconnect:

- *0b00*: The functionality is disabled
- From *0b01* to 2'b11: Number of stalls

Bit 8 - **HCI_HWPE_PRIO** *(R/W)*

HCI HWPE priority configuration bitfield:

- *0b0*: Core requests takes precedence over HWPE requests
- *0b1*: HWPE requests takes precedence over Core requests

8.1.1.2.4 Cluster clock gate configuration register. (CLOCK_GATE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN

Bit 0 - **EN** *(R/W)*

Cluster clock gate configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.1.2.5 Cluster cores debug resume register. (DBG_RESUME)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CORE7	CORE6	CORE5	CORE4	CORE3	CORE2	CORE1	CORE0

Bit 7 - **CORE7** *(W)*

Core 7 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 7

Bit 6 - **CORE6** *(W)*

Core 6 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 6

Bit 5 - **CORE5** *(W)*

Core 5 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 5

Bit 4 - **CORE4** *(W)*

Core 4 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 4

Bit 3 - **CORE3** *(W)*

Core 3 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 3

Bit 2 - **CORE2** *(W)*

Core 2 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 2

Bit 1 - **CORE1** *(W)*

Core 1 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 1

Bit 0 - **CORE0** *(W)*

Core 0 debug resume configuration bitfield:

- *0b0*: stay halted
- *0b1*: resume core 0

8.1.1.2.6 Cluster cores debug halt status register. (DBG_HALT_STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CORE7	CORE6	CORE5	CORE4	CORE3	CORE2	CORE1	CORE0

Bit 7 - **CORE7** *(R)*

Core 7 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 6 - **CORE6** *(R)*

Core 6 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 5 - **CORE5** *(R)*

Core 5 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 4 - **CORE4** *(R)*

Core 4 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 3 - **CORE3** *(R)*

Core 3 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 2 - **CORE2** *(R)*

Core 2 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 1 - **CORE1** *(R)*

Core 1 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

Bit 0 - **CORE0** *(R)*

Core 0 debug halt status flag bitfield:

- *0b0*: running
- *0b1*: halted

8.1.1.2.7 Cluster cores debug halt mask configuration register. (DBG_HALT_MASK)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CORE7	CORE6	CORE5	CORE4	CORE3	CORE2	CORE1	CORE0

Bit 7 - **CORE7** *(R/W)*

Core 7 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 6 - **CORE6** *(R/W)*

Core 6 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 5 - **CORE5** *(R/W)*

Core 5 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 4 - **CORE4** *(R/W)*

Core 4 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 3 - **CORE3** *(R/W)*

Core 3 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 2 - **CORE2** *(R/W)*

Core 2 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 1 - **CORE1** *(R/W)*

Core 1 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

Bit 0 - **CORE0** *(R/W)*

Core 0 debug halt mask bitfield. When bit is set, core will be part of mask group and stopped when one of the members of the group stops.

8.1.1.2.8 Cluster core 0 boot address configuration register. (BOOT_ADDR0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA															

Bits 31:0 - **BA** *(R/W)*

Cluster core 0 boot address configuration bitfield.

8.1.1.2.9 TCDM arbitration policy ch0 for cluster cores configuration register. (TCDM_ARB_POLICY_CH0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															POL

Bit 0 - **POL** *(R/W)*

TCDM arbitration policy for cluster cores configuration bitfield:

- *0b0*: fair round robin
- *0b1*: fixed order

8.1.1.2.10 TCDM arbitration policy ch1 for DMA/HWCE configuration register. (TCDM_ARB_POLICY_CH1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															POL

Bit 0 - **POL** *(R/W)*

TCDM arbitration policy for DMA/HWCE configuration bitfield:

- *0b0*: fair round robin
- *0b1*: fixed order

8.1.1.2.11 Read only duplicate of TCDM_ARB_POLICY_CH0 register (TCDM_ARB_POLICY_CH0_REP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															POL

Bit 0 - **POL** *(R/W)*

TCDM arbitration policy for cluster cores configuration bitfield:

- *0b0*: fair round robin
- *0b1*: fixed order

8.1.1.2.12 Read only duplicate of TCDM_ARB_POLICY_CH1 register (TCDM_ARB_POLICY_CH1_REP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															POL

Bit 0 - **POL** *(R/W)*

TCDM arbitration policy for DMA/HWCE configuration bitfield:

- *0b0*: fair round robin
- *0b1*: fixed order

8.1.2 Basic timer

None

8.1.2.1 Cluster timer registers

Name	Address	Size	Type	Access	Default	Description
CFG_LO	0x10200400	32	Config	R/W	*0x0000*	Timer Low Configuration register.
CFG_HI	0x10200404	32	Config	R/W	*0x0000*	Timer High Configuration register.
CNT_LO	0x10200408	32	Data	R/W	*0x0000*	Timer Low counter value register.
CNT_HI	0x1020040C	32	Data	R/W	*0x0000*	Timer High counter value register.
CMP_LO	0x10200410	32	Config	R/W	*0x0000*	Timer Low comparator value register.
CMP_HI	0x10200414	32	Config	R/W	*0x0000*	Timer High comparator value register.
START_LO	0x10200418	32	Config	R/W	*0x0000*	Start Timer Low counting register.
START_HI	0x1020041C	32	Config	R/W	*0x0000*	Start Timer High counting register.
RESET_LO	0x10200420	32	Config	R/W	*0x0000*	Reset Timer Low counter register.
RESET_HI	0x10200424	32	Config	R/W	*0x0000*	Reset Timer High counter register.

Table 6. Cluster timer registers table

8.1.2.2 Basic timer registers details

8.1.2.2.1 Timer Low Configuration register. (CFG_LO)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CASC	Reserved														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PVAL								CCFG	PEN	ONE_S	MODE	IEM	IRQEN	RESET	ENABLE

Bit 31 - **CASC** *(R/W)*

Timer low + Timer high 64bit cascaded mode configuration bitfield.

Bits 15:8 - **PVAL** *(R/W)*

Timer low prescaler value bitfield. Ftimer = Fclk / (1 + PRESC_VAL)

Bit 7 - **CCFG** *(R/W)*

Timer low clock source configuration bitfield:

- *0b0*: FLL or FLL+Prescaler
- *0b1*: Reference clock at 32kHz

Bit 6 - **PEN** *(R/W)*

Timer low prescaler enable configuration bitfield:- *0b0*: disabled

- *0b1*: enabled

Bit 5 - **ONE_S** *(R/W)*

Timer low one shot configuration bitfield:

- *0b0*: let Timer low enabled counting when compare match with CMP_LO occurs.
- *0b1*: disable Timer low when compare match with CMP_LO occurs.

Bit 4 - **MODE** *(R/W)*

Timer low continuous mode configuration bitfield:

- *0b0*: Continue mode - continue incrementing Timer low counter when compare match with CMP_LO occurs.
- *0b1*: Cycle mode - reset Timer low counter when compare match with CMP_LO occurs.

Bit 3 - **IEM** *(R/W)*

Timer low input event mask configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 2 - **IRQEN** *(R/W)*

Timer low compare match interrupt enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 1 - **RESET** *(R/W)*

Timer low counter reset command bitfield. Cleared after Timer Low reset execution.

Bit 0 - **ENABLE** *(R/W)*

Timer low enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.2.2.2 Timer High Configuration register. (CFG_HI)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CLKCFG	PEN	ONE_S	MODE	IEM	IRQEN	RESET	ENABLE

Bit 7 - **CLKCFG** *(R/W)*

Timer high clock source configuration bitfield:

- *0b0*: FLL or FLL+Prescaler
- *0b1*: Reference clock at 32kHz

Bit 6 - **PEN** *(R/W)*

Timer high prescaler enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 5 - **ONE_S** *(R/W)*

Timer high one shot configuration bitfield:

- *0b0*: let Timer high enabled counting when compare match with CMP_LO occurs.
- *0b1*: disable Timer high when compare match with CMP_LO occurs.

Bit 4 - **MODE** *(R/W)*

Timer high continuous mode configuration bitfield:

- *0b0*: Continue mode - continue incrementing Timer high counter when compare match with CMP_LO occurs.
- *0b1*: Cycle mode - reset Timer high counter when compare match with CMP_LO occurs.

Bit 3 - **IEM** *(R/W)*

Timer high input event mask configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 2 - **IRQEN** *(R/W)*

Timer high compare match interrupt enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 1 - **RESET** *(W)*

Timer high counter reset command bitfield. Cleared after Timer high reset execution.

Bit 0 - **ENABLE** *(R/W)*

Timer high enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.2.2.3 Timer Low counter value register. (CNT_LO)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT_LO															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT_LO															

Bits 31:0 - **CNT_LO** *(R/W)*

Timer Low counter value bitfield.

8.1.2.2.4 Timer High counter value register. (CNT_HI)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT_HI															

Bits 31:0 - **CNT_HI** *(R/W)*

Timer High counter value bitfield.

8.1.2.2.5 Timer Low comparator value register. (CMP_LO)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP_LO															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP_LO															

Bits 31:0 - **CMP_LO** *(R/W)*

Timer Low comparator value bitfield.

8.1.2.2.6 Timer High comparator value register. (CMP_HI)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP_HI															

Bits 31:0 - **CMP_HI** *(R/W)*

Timer High comparator value bitfield.

8.1.2.2.7 Start Timer Low counting register. (START_LO)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															STRT_LO

Bit 0 - **STRT_LO** *(W)*

Timer Low start command bitfield. When executed, CFG_LO.ENABLE is set.

8.1.2.2.8 Start Timer High counting register. (START_HI)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STRT_HI

Bit 0 - **STRT_HI** *(W)*

Timer High start command bitfield. When executed, CFG_HI.ENABLE is set.

8.1.2.2.9 Reset Timer Low counter register. (RESET_LO)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RST_LO

Bit 0 - **RST_LO** *(W)*

Timer Low counter reset command bitfield. When executed, CFG_LO.RESET is set.

8.1.2.2.10 Reset Timer High counter register. (RESET_HI)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RST_HI

Bit 0 - **RST_HI** *(W)*

Timer High counter reset command bitfield. When executed, CFG_HI.RESET is set.

8.1.3 CL_EVENT_UNIT

None

8.1.3.1 Cluster event unit registers

Name	Address	Size	Type	Access	Default	Description
EVT_MASK_CORE0	0x10200800	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE0	0x10200804	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE0	0x10200808	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE0	0x1020080C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE0	0x10200810	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE0	0x10200814	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE0	0x10200818	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE0	0x1020081C	32	Config	R	*0x0000*	Pending input events status register.

Name	Address	Size	Type	Access	Default	Description
EVENT_BUFFER_MASKED_CORE0	0x10200820	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE0	0x10200824	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE0	0x10200828	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE0	0x1020082C	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE0	0x10200830	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE0	0x10200834	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE1	0x10200840	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE1	0x10200844	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE1	0x10200848	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE1	0x1020084C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE1	0x10200850	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE1	0x10200854	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE1	0x10200858	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE1	0x1020085C	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE1	0x10200860	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE1	0x10200864	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE1	0x10200868	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE1	0x1020086C	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE1	0x10200870	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE1	0x10200874	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE2	0x10200880	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE2	0x10200884	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE2	0x10200888	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE2	0x1020088C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE2	0x10200890	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE2	0x10200894	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE2	0x10200898	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE2	0x1020089C	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE2	0x102008A0	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.

Name	Address	Size	Type	Access	Default	Description
EVENT_BUFFER_IRQ_MASKED_CORE2	0x102008A4	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE2	0x102008A8	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE2	0x102008AC	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE2	0x102008B0	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE2	0x102008B4	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE3	0x102008C0	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE3	0x102008C4	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE3	0x102008C8	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE3	0x102008CC	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE3	0x102008D0	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE3	0x102008D4	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE3	0x102008D8	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE3	0x102008DC	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE3	0x102008E0	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE3	0x102008E4	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE3	0x102008E8	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE3	0x102008EC	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE3	0x102008F0	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE3	0x102008F4	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE4	0x10200900	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE4	0x10200904	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE4	0x10200908	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE4	0x1020090C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE4	0x10200910	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE4	0x10200914	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE4	0x10200918	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE4	0x1020091C	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE4	0x10200920	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE4	0x10200924	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.

Name	Address	Size	Type	Access	Default	Description
EVENT_BUFFER_CLEAR_CORE4	0x10200928	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE4	0x1020092C	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE4	0x10200930	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE4	0x10200934	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE5	0x10200940	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE5	0x10200944	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE5	0x10200948	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE5	0x1020094C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE5	0x10200950	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE5	0x10200954	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE5	0x10200958	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE5	0x1020095C	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE5	0x10200960	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE5	0x10200964	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE5	0x10200968	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE5	0x1020096C	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE5	0x10200970	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE5	0x10200974	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE6	0x10200980	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE6	0x10200984	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE6	0x10200988	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE6	0x1020098C	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE6	0x10200990	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE6	0x10200994	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE6	0x10200998	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE6	0x1020099C	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE6	0x102009A0	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE6	0x102009A4	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE6	0x102009A8	32	Config	W	*0x0000*	Pending input events status clear command register.

Name	Address	Size	Type	Access	Default	Description
SW_EVENT_MASK_CORE6	0x102009AC	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE6	0x102009B0	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE6	0x102009B4	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
EVT_MASK_CORE7	0x102009C0	32	Config	R/W	*0x0000*	Input event mask configuration register.
EVT_MASK_AND_CORE7	0x102009C4	32	Config	W	*0x0000*	Input event mask update command register with bitwise AND operation.
EVT_MASK_OR_CORE7	0x102009C8	32	Config	W	*0x0000*	Input event mask update command register with bitwise OR operation.
IRQ_MASK_CORE7	0x102009CC	32	Config	R/W	*0x0000*	Interrupt request mask configuration register.
IRQ_MASK_AND_CORE7	0x102009D0	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise AND operation.
IRQ_MASK_OR_CORE7	0x102009D4	32	Config	W	*0x0000*	Interrupt request mask update command register with bitwise OR operation.
CLOCK_STATUS_CORE7	0x102009D8	32	Config	R	*0x0000*	Cluster cores clock status register.
EVENT_BUFFER_CORE7	0x102009DC	32	Config	R	*0x0000*	Pending input events status register.
EVENT_BUFFER_MASKED_CORE7	0x102009E0	32	Config	R	*0x0000*	Pending input events status register with EVT_MASK applied.
EVENT_BUFFER_IRQ_MASKED_CORE7	0x102009E4	32	Config	R	*0x0000*	Pending input events status register with IRQ_MASK applied.
EVENT_BUFFER_CLEAR_CORE7	0x102009E8	32	Config	W	*0x0000*	Pending input events status clear command register.
SW_EVENT_MASK_CORE7	0x102009EC	32	Config	R/W	*0x0000*	Software events cluster cores destination mask configuration register.
SW_EVENT_MASK_AND_CORE7	0x102009F0	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise AND operation.
SW_EVENT_MASK_OR_CORE7	0x102009F4	32	Config	W	*0x0000*	Software events cluster cores destination mask update command register with bitwise OR operation.
HW_BARRIER_0_TRIG_MASK	0x10200C00	32	Config	R/W	*0x0000*	Cluster hardware barrier 0 trigger mask configuration register.
HW_BARRIER_0_STATUS	0x10200C04	32	Status	R	*0x0000*	Cluster hardware barrier 0 status register.
HW_BARRIER_0_STATUS_SUM	0x10200C08	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_0_TARGET_MASK	0x10200C0C	32	Config	R/W	*0x0000*	Cluster hardware barrier 0 target mask configuration register.
HW_BARRIER_0_TRIG	0x10200C10	32	Config	W	*0x0000*	Cluster hardware barrier 0 trigger command register.
HW_BARRIER_1_TRIG_MASK	0x10200C20	32	Config	R/W	*0x0000*	Cluster hardware barrier 1 trigger mask configuration register.
HW_BARRIER_1_STATUS	0x10200C24	32	Status	R	*0x0000*	Cluster hardware barrier 1 status register.
HW_BARRIER_1_STATUS_SUM	0x10200C28	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_1_TARGET_MASK	0x10200C2C	32	Config	R/W	*0x0000*	Cluster hardware barrier 1 target mask configuration register.
HW_BARRIER_1_TRIG	0x10200C30	32	Config	W	*0x0000*	Cluster hardware barrier 1 trigger command register.
HW_BARRIER_2_TRIG_MASK	0x10200C40	32	Config	R/W	*0x0000*	Cluster hardware barrier 2 trigger mask configuration register.
HW_BARRIER_2_STATUS	0x10200C44	32	Status	R	*0x0000*	Cluster hardware barrier 2 status register.

Name	Address	Size	Type	Access	Default	Description
HW_BARRIER_2_STATUS_SUM	0x10200C48	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_2_TARGET_MASK	0x10200C4C	32	Config	R/W	*0x0000*	Cluster hardware barrier 2 target mask configuration register.
HW_BARRIER_2_TRIG	0x10200C50	32	Config	W	*0x0000*	Cluster hardware barrier 2 trigger command register.
HW_BARRIER_3_TRIG_MASK	0x10200C60	32	Config	R/W	*0x0000*	Cluster hardware barrier 3 trigger mask configuration register.
HW_BARRIER_3_STATUS	0x10200C64	32	Status	R	*0x0000*	Cluster hardware barrier 3 status register.
HW_BARRIER_3_STATUS_SUM	0x10200C68	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_3_TARGET_MASK	0x10200C6C	32	Config	R/W	*0x0000*	Cluster hardware barrier 3 target mask configuration register.
HW_BARRIER_3_TRIG	0x10200C70	32	Config	W	*0x0000*	Cluster hardware barrier 3 trigger command register.
HW_BARRIER_4_TRIG_MASK	0x10200C80	32	Config	R/W	*0x0000*	Cluster hardware barrier 4 trigger mask configuration register.
HW_BARRIER_4_STATUS	0x10200C84	32	Status	R	*0x0000*	Cluster hardware barrier 4 status register.
HW_BARRIER_4_STATUS_SUM	0x10200C88	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_4_TARGET_MASK	0x10200C8C	32	Config	R/W	*0x0000*	Cluster hardware barrier 4 target mask configuration register.
HW_BARRIER_4_TRIG	0x10200C90	32	Config	W	*0x0000*	Cluster hardware barrier 4 trigger command register.
HW_BARRIER_5_TRIG_MASK	0x10200CA0	32	Config	R/W	*0x0000*	Cluster hardware barrier 5 trigger mask configuration register.
HW_BARRIER_5_STATUS	0x10200CA4	32	Status	R	*0x0000*	Cluster hardware barrier 5 status register.
HW_BARRIER_5_STATUS_SUM	0x10200CA8	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_5_TARGET_MASK	0x10200CAC	32	Config	R/W	*0x0000*	Cluster hardware barrier 5 target mask configuration register.
HW_BARRIER_5_TRIG	0x10200CB0	32	Config	W	*0x0000*	Cluster hardware barrier 5 trigger command register.
HW_BARRIER_6_TRIG_MASK	0x10200CC0	32	Config	R/W	*0x0000*	Cluster hardware barrier 6 trigger mask configuration register.
HW_BARRIER_6_STATUS	0x10200CC4	32	Status	R	*0x0000*	Cluster hardware barrier 6 status register.
HW_BARRIER_6_STATUS_SUM	0x10200CC8	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_6_TARGET_MASK	0x10200CCC	32	Config	R/W	*0x0000*	Cluster hardware barrier 6 target mask configuration register.
HW_BARRIER_6_TRIG	0x10200CD0	32	Config	W	*0x0000*	Cluster hardware barrier 6 trigger command register.
HW_BARRIER_7_TRIG_MASK	0x10200CE0	32	Config	R/W	*0x0000*	Cluster hardware barrier 7 trigger mask configuration register.
HW_BARRIER_7_STATUS	0x10200CE4	32	Status	R	*0x0000*	Cluster hardware barrier 7 status register.
HW_BARRIER_7_STATUS_SUM	0x10200CE8	32	Status	R	*0x0000*	Cluster hardware barrier summary status register.
HW_BARRIER_7_TARGET_MASK	0x10200CEC	32	Config	R/W	*0x0000*	Cluster hardware barrier 7 target mask configuration register.
HW_BARRIER_7_TRIG	0x10200CF0	32	Config	W	*0x0000*	Cluster hardware barrier 7 trigger command register.
SW_EVENT_0_TRIG	0x10200E00	32	Config	W	*0x0000*	Cluster Software event 0 trigger command register.
SW_EVENT_1_TRIG	0x10200E04	32	Config	W	*0x0000*	Cluster Software event 1 trigger command register.

Name	Address	Size	Type	Access	Default	Description
SW_EVENT_2_TRIG	0x10200E08	32	Config	W	*0x0000*	Cluster Software event 2 trigger command register.
SW_EVENT_3_TRIG	0x10200E0C	32	Config	W	*0x0000*	Cluster Software event 3 trigger command register.
SW_EVENT_4_TRIG	0x10200E10	32	Config	W	*0x0000*	Cluster Software event 4 trigger command register.
SW_EVENT_5_TRIG	0x10200E14	32	Config	W	*0x0000*	Cluster Software event 5 trigger command register.
SW_EVENT_6_TRIG	0x10200E18	32	Config	W	*0x0000*	Cluster Software event 6 trigger command register.
SW_EVENT_7_TRIG	0x10200E1C	32	Config	W	*0x0000*	Cluster Software event 7 trigger command register.
SOC_PERIPH_EVENT_ID	0x10200F00	32	Status	R	*0x0000*	Cluster SoC peripheral event ID status register.

Table 7. Cluster event unit registers table

8.1.3.2 CL_EVENT_UNIT registers details

8.1.3.2.1 Input event mask configuration register. (EVT_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMSOC	Reserved	EMCL													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EMCL															

Bit 31 - **EMSOC** *(R/W)*

Soc peripheral input event mask configuration bitfield:

- EMSOC[i]=*0b0*: Input event request i is masked
- EMSOC[i]=*0b1*: Input event request i is not masked

Bits 29:0 - **EMCL** *(R/W)*

Cluster internal input event mask configuration bitfield:

- EMCL[i]=*0b0*: Input event request i is masked
- EMCL[i]=*0b1*: Input event request i is not masked

8.1.3.2.2 Hardware task dispatcher push command register. (HW_DISPATCH_PUSH_TASK)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSG															

Bits 31:0 - **MSG** *(W)*

Message to dispatch to all cluster cores selected in HW_DISPATCH_PUSH_TEAM_CONFIG.CT configuration bitfield.

8.1.3.2.3 Hardware task dispatcher pop command register. (HW_DISPATCH_POP_TASK)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSG															

Bits 31:0 - **MSG** *(R)*

Message dispatched using HW_DISPATCH_PUSH_TASK command and popped by cluster core who issued HW_DISPATCH_POP_TASK command.

8.1.3.2.4 Hardware mutex 0 non-blocking put command register. (HW_MUTEX_0_MSG_PUT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSG															

Bits 31:0 - **MSG** *(W)*

Message pushed when releasing hardware mutex 0 configuration bitfiled. It is a non-blocking access.

8.1.3.2.5 Hardware mutex 0 blocking get command register. (HW_MUTEX_0_MSG_GET)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSG															

Bits 31:0 - **MSG** *(R)*

Message popped when taking hardware mutex 0 data bitfiled. It is a blocking access.

8.1.3.2.6 Cluster Software event 0 trigger command register. (SW_EVENT_0_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW0T							

Bits 7:0 - **SW0T** *(W)*

Triggers software event 0 for cluster core i if SW0T[i]=*0b1*.

8.1.3.2.7 Cluster Software event 0 trigger and wait command register. (SW_EVENT_0_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 0 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.8 Cluster Software event 0 trigger, wait and clear command register. (SW_EVENT_0_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 0 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.9 Cluster SoC peripheral event ID status register. (SOC_PERIPH_EVENT_ID)

Reset value: 0x0000

Host access bus: PERIPH

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID	Reserved														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								ID							

Bit 31 - **VALID** *(R)*

Validity bit of SOC_PERIPH_EVENT_ID.ID bitfield.

Bits 7:0 - **ID** *(R)*

Oldest SoC peripheral event ID status bitfield.

8.1.3.2.10 Cluster hardware barrier 0 trigger mask configuration register. (HW_BARRIER_0_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HB0TM							

Bits 7:0 - **HB0TM** *(R/W)*

Trigger mask for hardware barrier 0 bitfield. Hardware barrier 0 will be triggered only if for all HB0TM[i] =*0b1*, HW_BARRIER_0_STATUS.HB0S[i]=*0b1*. HB0TM=0 means that hardware barrier 0 is disabled.

8.1.3.2.11 Input event mask update command register with bitwise AND operation. (EVT_MASK_AND)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMA															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EMA															

Bits 31:0 - **EMA** *(W)*

Input event mask configuration bitfield update with bitwise AND operation. It allows clearing EMCL[i], EMINTCL[i] or EMSOC[i] if EMA[i]=*0b1*.

8.1.3.2.12 Hardware task dispatcher cluster core team configuration register. (HW_DISPATCH_PUSH_TEAM_CONFIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								CT							

Bits 7:0 - **CT** *(R/W)*

Cluster cores team selection configuration bitfield. It allows to transmit HW_DISPATCH_PUSH_TASK.MSG to cluster core i if CT[i]=*0b1*.

8.1.3.2.13 Hardware mutex 1 non-blocking put command register. (HW_MUTEX_1_MSG_PUT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSG															

Bits 31:0 - **MSG** *(W)*

Message pushed when releasing hardware mutex 1 configuration bitfiled. It is a non-blocking access.

8.1.3.2.14 Hardware mutex 1 blocking get command register. (HW_MUTEX_1_MSG_GET)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSG															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSG															

Bits 31:0 - **MSG** *(R)*

Message popped when taking hardware mutex 1 data bitfiled. It is a blocking access.

8.1.3.2.15 Cluster Software event 1 trigger command register. (SW_EVENT_1_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								SW1T							

Bits 7:0 - **SW1T** *(W)*

Triggers software event 1 for cluster core i if SW1T[i]=*0b1*.

8.1.3.2.16 Cluster Software event 1 trigger and wait command register. (SW_EVENT_1_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 1 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.17 Cluster Software event 1 trigger, wait and clear command register. (SW_EVENT_1_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 1 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.18 Cluster hardware barrier 0 status register. (HW_BARRIER_0_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 0 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 0. It is cleared when HBS matches HW_BARRIER_0_TRIG_MASK.HB0TM.

8.1.3.2.19 Input event mask update command register with bitwise OR operation. (EVT_MASK_OR)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMO															

Bits 31:0 - **EMO** *(W)*

Input event mask configuration bitfield update with bitwise OR operation. It allows setting EMCL[i], EMINTCL[i] or EMSOC[i] if EMO[i]=*0b1*.

8.1.3.2.20 Cluster Software event 2 trigger command register. (SW_EVENT_2_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW2T							

Bits 7:0 - **SW2T** *(W)*

Triggers software event 2 for cluster core i if SW2T[i]=*0b1*.

8.1.3.2.21 Cluster Software event 2 trigger and wait command register. (SW_EVENT_2_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 2 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.22 Cluster Software event 2 trigger, wait and clear command register. (SW_EVENT_2_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 2 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.23 Cluster hardware barrier summary status register. (HW_BARRIER_0_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 0. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.24 Interrupt request mask configuration register. (IRQ_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMSOC	IMINTCL	IMCL													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMCL															

Bit 31 - **IMSOC** *(R/W)*

Soc peripheral interrupt request mask configuration bitfield:

- bit[i]=*0b0*: Interrupt request i is masked
- bit[i]=*0b1*: Interrupt request i is not masked

Bit 30 - **IMINTCL** *(R/W)*

Inter-cluster interrupt request mask configuration bitfield:

- bit[i]=*0b0*: Interrupt request i is masked
- bit[i]=*0b1*: Interrupt request i is not masked

Bits 29:0 - **IMCL** *(R/W)*

Cluster internal interrupt request mask configuration bitfield:

- bit[i]=*0b0*: Interrupt request i is masked
- bit[i]=*0b1*: Interrupt request i is not masked

8.1.3.2.25 Cluster Software event 3 trigger command register. (SW_EVENT_3_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW3T							

Bits 7:0 - **SW3T** *(W)*

Triggers software event 3 for cluster core i if SW3T[i]=*0b1*.

8.1.3.2.26 Cluster Software event 3 trigger and wait command register. (SW_EVENT_3_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 3 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.27 Cluster Software event 3 trigger, wait and clear command register. (SW_EVENT_3_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 3 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.28 Cluster hardware barrier 0 target mask configuration register. (HW_BARRIER_0_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 0 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 0 event when HW_BARRIER_0_STATUS will match HW_BARRIER_0_TRIG_MASK.

8.1.3.2.29 Interrupt request mask update command register with bitwise AND operation. (IRQ_MASK_AND)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMA															

Bits 31:0 - **IMA** *(W)*

Interrupt request mask configuration bitfield update with bitwise AND operation. It allows clearing IMCL[i], IMINTCL[i] or IMSOC[i] if IMA[i]=*0b1*.

8.1.3.2.30 Cluster Software event 4 trigger command register. (SW_EVENT_4_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW4T							

Bits 7:0 - **SW4T** *(W)*

Triggers software event 4 for cluster core i if SW4T[i]=*0b1*.

8.1.3.2.31 Cluster Software event 4 trigger and wait command register. (SW_EVENT_4_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 4 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.32 Cluster Software event 4 trigger, wait and clear command register. (SW_EVENT_4_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 4 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.33 Cluster hardware barrier 0 trigger command register. (HW_BARRIER_0_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_0_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.34 Interrupt request mask update command register with bitwise OR operation. (IRQ_MASK_OR)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMO															

Bits 31:0 - **IMO** *(W)*

Interrupt request mask configuration bitfield update with bitwise OR operation. It allows setting IMCL[i], IMINTCL[i] or IMSOC[i] if IMO[i]=*0b1*.

8.1.3.2.35 Cluster Software event 5 trigger command register. (SW_EVENT_5_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW5T							

Bits 7:0 - **SW5T** *(W)*

Triggers software event 5 for cluster core i if SW5T[i]=*0b1*.

8.1.3.2.36 Cluster Software event 5 trigger and wait command register. (SW_EVENT_5_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 5 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.37 Cluster Software event 5 trigger, wait and clear command register. (SW_EVENT_5_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 5 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.38 Cluster hardware barrier 0 self trigger command register. (HW_BARRIER_0_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_0_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.39 Cluster cores clock status register. (CLOCK_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CS

Bit 0 - **CS** *(R)*

Cluster core clock status bitfield:

- *0b0*: Cluster core clocked is gated
- *0b1*: Cluster core clocked is running

8.1.3.2.40 Cluster Software event 6 trigger command register. (SW_EVENT_6_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW6T							

Bits 7:0 - **SW6T** *(W)*

Triggers software event 6 for cluster core i if SW6T[i]=*0b1*.

8.1.3.2.41 Cluster Software event 6 trigger and wait command register. (SW_EVENT_6_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 6 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.42 Cluster Software event 6 trigger, wait and clear command register. (SW_EVENT_6_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 6 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.43 Cluster hardware barrier 0 trigger and wait command register. (HW_BARRIER_0_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_0[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_0 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.44 Pending input events status register. (EVENT_BUFFER)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EB															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EB															

Bits 31:0 - **EB** *(R)*

Pending input events status bitfield.

EB[i]=*0b1*: one or more input event i request are pending.

8.1.3.2.45 Cluster Software event 7 trigger command register. (SW_EVENT_7_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SW7T							

Bits 7:0 - **SW7T** *(W)*

Triggers software event 7 for cluster core i if SW7T[i]=*0b1*.

8.1.3.2.46 Cluster Software event 7 trigger and wait command register. (SW_EVENT_7_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 7 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.47 Cluster Software event 7 trigger, wait and clear command register. (SW_EVENT_7_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Triggers software event 7 to all cluster cores targeted in SW_EVENT_MASK and gate the issuing cluster core clock until at least one unmasked event occurs. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.48 Cluster hardware barrier 0 trigger, wait and clear command register. (HW_BARRIER_0_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_0[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_0 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.49 Pending input events status register with EVT_MASK applied. (EVENT_BUFFER_MASKED)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Pending input events status bitfield with EVT_MASK applied.

EBM[i]=*0b1*: one or more input event i request are pending.

8.1.3.2.50 Cluster hardware barrier 1 trigger mask configuration register. (HW_BARRIER_1_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB1TM							

Bits 7:0 - **HB1TM** *(R/W)*

Trigger mask for hardware barrier 1 bitfield. Hardware barrier 1 will be triggered only if for all HB1TM[i] =*0b1*, HW_BARRIER_1_STATUS.HB1S[i]=*0b1*. HB1TM=0 means that hardware barrier 1 is disabled.

8.1.3.2.51 Pending input events status register with IRQ_MASK applied. (EVENT_BUFFER_IRQ_MASKED)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBM															

Bits 31:0 - **IBM** *(R)*

Pending input events status bitfield with IRQ_MASK applied.

IBM[i]=*0b1*: one or more input events i are pending.

8.1.3.2.52 Cluster hardware barrier 1 status register. (HW_BARRIER_1_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 1 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 1. It is cleared when HBS matches HW_BARRIER_1_TRIG_MASK.HB1TM.

8.1.3.2.53 Pending input events status clear command register. (EVENT_BUFFER_CLEAR)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBC															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBC															

Bits 31:0 - **EBC** *(W)*

Pending input events status clear command bitfield. It allows clearing EB[i] if EBC[i]=*0b1*.

8.1.3.2.54 Cluster hardware barrier summary status register. (HW_BARRIER_1_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 1. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.55 Software events cluster cores destination mask configuration register. (SW_EVENT_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SWEM							

Bits 7:0 - **SWEM** *(R/W)*

Software events mask configuration bitfield:

- bit[i]=*0b0*: software events are masked for CL_CORE[i]
- bit[i]=*0b1*: software events are not masked for CL_CORE[i]

8.1.3.2.56 Cluster hardware barrier 1 target mask configuration register. (HW_BARRIER_1_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 1 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 1 event when HW_BARRIER_1_STATUS will match HW_BARRIER_1_TRIG_MASK.

8.1.3.2.57 Software events cluster cores destination mask update command register with bitwise AND operation. (SW_EVENT_MASK_AND)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SWEMA							

Bits 7:0 - **SWEMA** *(W)*

Software event mask configuration bitfield update with bitwise AND operation. It allows clearing SWEM[i] if SWEMA[i]=*0b1*.

8.1.3.2.58 Cluster hardware barrier 1 trigger command register. (HW_BARRIER_1_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_1_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.59 Software events cluster cores destination mask update command register with bitwise OR operation. (SW_EVENT_MASK_OR)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SWEMO							

Bits 7:0 - **SWEMO** *(W)*

Software event mask configuration bitfield update with bitwise OR operation. It allows setting SWEM[i] if SWEMO[i]=*0b1*.

8.1.3.2.60 Cluster hardware barrier 1 self trigger command register. (HW_BARRIER_1_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_1_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.61 Input event wait command register. (EVENT_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Reading this register will gate the Cluster core clock until at least one unmasked event occurs. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.62 Cluster hardware barrier 1 trigger and wait command register. (HW_BARRIER_1_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_1[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_1 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.63 Input event wait and clear command register. (EVENT_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Reading this register has the same effect as reading EVENT_WAIT.EBM. In addition, EVENT_BUFFER.EB[i] bits are cleared if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.64 Cluster hardware barrier 1 trigger, wait and clear command register. (HW_BARRIER_1_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_1[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_1 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.65 Cluster hardware barrier 2 trigger mask configuration register. (HW_BARRIER_2_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB2TM							

Bits 7:0 - **HB2TM** *(R/W)*

Trigger mask for hardware barrier 2 bitfield. Hardware barrier 2 will be triggered only if for all HB2TM[i] =*0b1*, HW_BARRIER_2_STATUS.HB2S[i]=*0b1*. HB2TM=0 means that hardware barrier 2 is disabled.

8.1.3.2.66 Cluster hardware barrier 2 status register. (HW_BARRIER_2_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 2 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 2. It is cleared when HBS matches HW_BARRIER_2_TRIG_MASK.HB2TM.

8.1.3.2.67 Cluster hardware barrier summary status register. (HW_BARRIER_2_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 2. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.68 Cluster hardware barrier 2 target mask configuration register. (HW_BARRIER_2_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 2 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 2 event when HW_BARRIER_2_STATUS will match HW_BARRIER_2_TRIG_MASK.

8.1.3.2.69 Cluster hardware barrier 2 trigger command register. (HW_BARRIER_2_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_2_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.70 Cluster hardware barrier 2 self trigger command register. (HW_BARRIER_2_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_2_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.71 Cluster hardware barrier 2 trigger and wait command register. (HW_BARRIER_2_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_2[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_2 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.72 Cluster hardware barrier 2 trigger, wait and clear command register. (HW_BARRIER_2_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_2[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_2 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.73 Cluster hardware barrier 3 trigger mask configuration register. (HW_BARRIER_3_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB3TM							

Bits 7:0 - **HB3TM** *(R/W)*

Trigger mask for hardware barrier 3 bitfield. Hardware barrier 3 will be triggered only if for all HB3TM[i] =*0b1*, HW_BARRIER_3_STATUS.HB3S[i]=*0b1*. HB3TM=0 means that hardware barrier 3 is disabled.

8.1.3.2.74 Cluster hardware barrier 3 status register. (HW_BARRIER_3_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 3 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 3. It is cleared when HBS matches HW_BARRIER_3_TRIG_MASK.HB3TM.

8.1.3.2.75 Cluster hardware barrier summary status register. (HW_BARRIER_3_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 3. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.76 Cluster hardware barrier 3 target mask configuration register. (HW_BARRIER_3_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 3 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 3 event when HW_BARRIER_3_STATUS will match HW_BARRIER_3_TRIG_MASK.

8.1.3.2.77 Cluster hardware barrier 3 trigger command register. (HW_BARRIER_3_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_3_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.78 Cluster hardware barrier 3 self trigger command register. (HW_BARRIER_3_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_3_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.79 Cluster hardware barrier 3 trigger and wait command register. (HW_BARRIER_3_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_3[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_3 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.80 Cluster hardware barrier 3 trigger, wait and clear command register. (HW_BARRIER_3_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_3[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_3 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.81 Cluster hardware barrier 4 trigger mask configuration register. (HW_BARRIER_4_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HB4TM							

Bits 7:0 - **HB4TM** *(R/W)*

Trigger mask for hardware barrier 4 bitfield. Hardware barrier 4 will be triggered only if for all HB4TM[i] =*0b1*, HW_BARRIER_4_STATUS.HB4S[i]=*0b1*. HB4TM=0 means that hardware barrier 4 is disabled.

8.1.3.2.82 Cluster hardware barrier 4 status register. (HW_BARRIER_4_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 4 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 4. It is cleared when HBS matches HW_BARRIER_4_TRIG_MASK.HB4TM.

8.1.3.2.83 Cluster hardware barrier summary status register. (HW_BARRIER_4_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 4. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.84 Cluster hardware barrier 4 target mask configuration register. (HW_BARRIER_4_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 4 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 4 event when HW_BARRIER_4_STATUS will match HW_BARRIER_4_TRIG_MASK.

8.1.3.2.85 Cluster hardware barrier 4 trigger command register. (HW_BARRIER_4_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_4_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.86 Cluster hardware barrier 4 self trigger command register. (HW_BARRIER_4_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_4_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.87 Cluster hardware barrier 4 trigger and wait command register. (HW_BARRIER_4_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_4[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_4 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.88 Cluster hardware barrier 4 trigger, wait and clear command register. (HW_BARRIER_4_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_4[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_4 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.89 Cluster hardware barrier 5 trigger mask configuration register. (HW_BARRIER_5_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB5TM							

Bits 7:0 - **HB5TM** *(R/W)*

Trigger mask for hardware barrier 5 bitfield. Hardware barrier 5 will be triggered only if for all HB5TM[i] =*0b1*, HW_BARRIER_5_STATUS.HB5S[i]=*0b1*. HB5TM=0 means that hardware barrier 5 is disabled.

8.1.3.2.90 Cluster hardware barrier 5 status register. (HW_BARRIER_5_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 5 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 5. It is cleared when HBS matches HW_BARRIER_5_TRIG_MASK.HB5TM.

8.1.3.2.91 Cluster hardware barrier summary status register. (HW_BARRIER_5_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 5. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.92 Cluster hardware barrier 5 target mask configuration register. (HW_BARRIER_5_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 5 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 5 event when HW_BARRIER_5_STATUS will match HW_BARRIER_5_TRIG_MASK.

8.1.3.2.93 Cluster hardware barrier 5 trigger command register. (HW_BARRIER_5_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_5_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.94 Cluster hardware barrier 5 self trigger command register. (HW_BARRIER_5_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_5_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.95 Cluster hardware barrier 5 trigger and wait command register. (HW_BARRIER_5_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_5[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_5 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.96 Cluster hardware barrier 5 trigger, wait and clear command register. (HW_BARRIER_5_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_5[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_5 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.97 Cluster hardware barrier 6 trigger mask configuration register. (HW_BARRIER_6_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB6TM							

Bits 7:0 - **HB6TM** *(R/W)*

Trigger mask for hardware barrier 6 bitfield. Hardware barrier 6 will be triggered only if for all HB6TM[i] =*0b1*, HW_BARRIER_6_STATUS.HB6S[i]=*0b1*. HB6TM=0 means that hardware barrier 6 is disabled.

8.1.3.2.98 Cluster hardware barrier 6 status register. (HW_BARRIER_6_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 6 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 6. It is cleared when HBS matches HW_BARRIER_6_TRIG_MASK.HB6TM.

8.1.3.2.99 Cluster hardware barrier summary status register. (HW_BARRIER_6_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 6. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.100 Cluster hardware barrier 6 target mask configuration register. (HW_BARRIER_6_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 6 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 6 event when HW_BARRIER_6_STATUS will match HW_BARRIER_6_TRIG_MASK.

8.1.3.2.101 Cluster hardware barrier 6 trigger command register. (HW_BARRIER_6_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_6_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.102 Cluster hardware barrier 6 self trigger command register. (HW_BARRIER_6_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_6_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.103 Cluster hardware barrier 6 trigger and wait command register. (HW_BARRIER_6_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_6[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_6 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.104 Cluster hardware barrier 6 trigger, wait and clear command register. (HW_BARRIER_6_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_6[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_6 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.105 Cluster hardware barrier 7 trigger mask configuration register. (HW_BARRIER_7_TRIG_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HB7TM							

Bits 7:0 - **HB7TM** *(R/W)*

Trigger mask for hardware barrier 7 bitfield. Hardware barrier 7 will be triggered only if for all HB7TM[i] =*0b1*, HW_BARRIER_7_STATUS.HB7S[i]=*0b1*. HB7TM=0 means that hardware barrier 7 is disabled.

8.1.3.2.106 Cluster hardware barrier 7 status register. (HW_BARRIER_7_STATUS)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBS							

Bits 7:0 - **HBS** *(R)*

Current status of hardware barrier 7 bitfield. HBS[i]=*0b1* means that cluster core i has triggered hardware barrier 7. It is cleared when HBS matches HW_BARRIER_7_TRIG_MASK.HB7TM.

8.1.3.2.107 Cluster hardware barrier summary status register. (HW_BARRIER_7_STATUS_SUM)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBSS							

Bits 7:0 - **HBSS** *(R)*

Current status of hardware barrier 7. HBSS[i] represents a summary of the barrier status for core i.

8.1.3.2.108 Cluster hardware barrier 7 target mask configuration register. (HW_BARRIER_7_TARGET_MASK)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								HBTAM							

Bits 7:0 - **HBTAM** *(R/W)*

Cluster hardware barrier 7 target mask configuration bitfield. HBATM[i]=*0b1* means that cluster core i will receive hardware barrier 7 event when HW_BARRIER_7_STATUS will match HW_BARRIER_7_TRIG_MASK.

8.1.3.2.109 Cluster hardware barrier 7 trigger command register. (HW_BARRIER_7_TRIG)

Reset value: 0x0000

Host access bus: PERIPH/DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								T							

Bits 7:0 - **T** *(W)*

Sets HW_BARRIER_7_STATUS.HBS[i] to *0b1* when T[i]=*0b1*.

8.1.3.2.110 Cluster hardware barrier 7 self trigger command register. (HW_BARRIER_7_SELF_TRIG)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T															

Bits 31:0 - **T** *(R)*

Sets HW_BARRIER_7_STATUS.HBS[i] to *0b1* when issued by cluster core i.

8.1.3.2.111 Cluster hardware barrier 7 trigger and wait command register. (HW_BARRIER_7_TRIG_WAIT)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_7[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_7 is released. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.3.2.112 Cluster hardware barrier 7 trigger, wait and clear command register. (HW_BARRIER_7_TRIG_WAIT_CLEAR)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBM															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBM															

Bits 31:0 - **EBM** *(R)*

Set HW_BARRIER_7[i] when issued by cluster core i and gate the issuing cluster core i clock until HW_BARRIER_7 is released. In addition, EVENT_BUFFER.EB[i] bits are cleared after the read if EVT_MASK[i]=*0b1*. The read content of this bitfield is equivalent to EVENT_BUFFER_MASKED.EBM

8.1.4 Cluster instruction cache control unit

None

8.1.4.1 Cluster icache control registers

Name	Address	Size	Type	Access	Default	Description
ENABLE	0x10201400	32	Config	W	*0x0000*	Cluster instruction cache unit enable configuration register.
FLUSH	0x10201404	32	Config	W	*0x0000*	Cluster instruction cache unit flush command register.
SEL_FLUSH	0x1020140C	32	Config	W	*0x0000*	Cluster instruction cache unit selective flush command register.
L1_L15_PREFETCH	0x1020141C	32	Config	W	*0x0000*	Enable L1 and L1.5 prefetch register.

Table 8. Cluster icache control registers table

8.1.4.2 Cluster instruction cache control unit registers details

8.1.4.2.1 Cluster instruction cache unit enable configuration register. (ENABLE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN

Bit 0 - **EN** *(R/W)*

Cluster instruction cache enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.4.2.2 Cluster instruction cache unit flush command register. (FLUSH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															FL

Bit 0 - **FL** *(R/W)*

Cluster instruction cache full flush command.

8.1.4.2.3 Cluster instruction cache unit selective flush command register. (SEL_FLUSH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															

Bits 31:0 - **ADDR** *(R/W)*

Cluster instruction cache selective flush address configuration bitfield.

8.1.4.2.4 Enable L1 and L1.5 prefetch register. (L1_L15_PREFETCH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CORE7	CORE6	CORE5	CORE4	CORE3	CORE2	CORE1	CORE0

Bit 7 - **CORE7** *(R/W)*

Core 7 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 6 - **CORE6** *(R/W)*

Core 6 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 5 - **CORE5** *(R/W)*

Core 5 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 4 - **CORE4** *(R/W)*

Core 4 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 3 - **CORE3** *(R/W)*

Core 3 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 2 - **CORE2** *(R/W)*

Core 2 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 1 - **CORE1** *(R/W)*

Core 1 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 0 - **CORE0** *(R/W)*

Core 0 icache prefetch enable configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.1.5 DMA

None

8.1.5.1 DMA registers

Name	Address	Size	Type	Access	Default	Description

Table 9. DMA registers table

8.1.5.2 DMA registers details

8.1.5.2.1 Cluster DMA configuration register. (CMD)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMD															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD															

Bits 31:0 - **CMD** *(R/W)*

Format is operation dependent. See below.

8.1.5.2.2 Cluster DMA status register. (STATUS)

Reset value: 0x0000

Host access bus: DEMUX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATUS															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STATUS															

Bits 31:0 - **STATUS** *(R/W)*

Format is operation dependent. See below.

8.1.5.3 DMA states

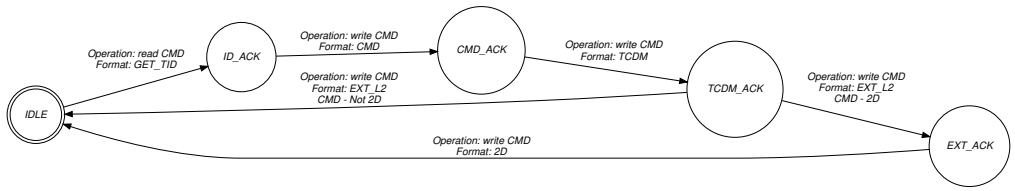

Figure 9. Queue transaction with ID

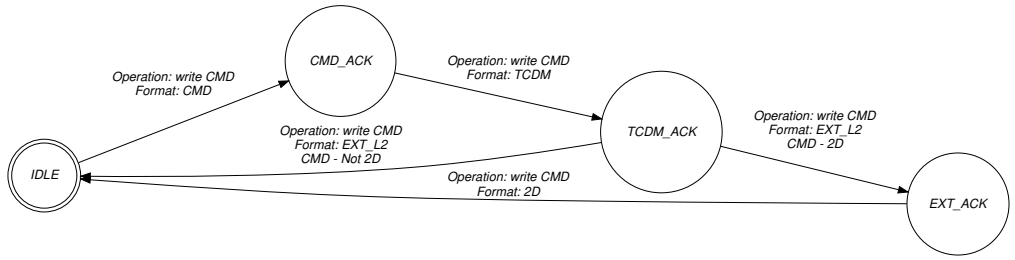

Figure 10. Queue transaction without ID

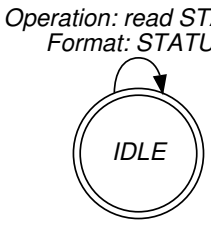

Figure 11. Get DMA status

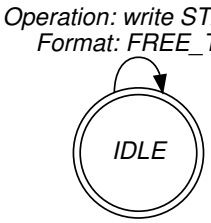

Figure 12. Free DMA transfer

8.1.5.4 DMA state command formats

Format Name	Register	Size	Access type	Description
GET_TID	CMD	32	R	Cluster DMA transfer identifier format.
CMD	CMD	32	W	Cluster DMA transfer configuration format.
STATUS	STATUS	32	R	Cluster DMA transfer free command format.
FREE_TID	STATUS	32	W	Cluster DMA transfer status format.
TCDM	CMD	32	W	Cluster DMA L1 base address configuration format.
EXT_L2	CMD	32	W	Cluster DMA L2 base address configuration format.
2D	CMD	32	W	Cluster DMA 2D transfer configuration format.

Table 10. DMA command format table

8.1.5.4.1 Cluster DMA transfer identifier format. (GET_TID)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TID			

Bits 3:0 - **TID** *(R)*

Transfer identifier value bitfield.

8.1.5.4.2 Cluster DMA transfer configuration format. (CMD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										BLE	ILE	ELE	2D	INC	TYPE
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEN															

Bit 21 - **BLE** *(W)*

Transfer event or interrupt broadcast configuration bitfield:

- *0b0*: event or interrupt is routed to the cluster core who initiated the transfer
- *0b1*: event or interrupt are broadcasted to all cluster cores

Bit 20 - **ILE** *(W)*

Transfer interrupt generation configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 19 - **ELE** *(W)*

Transfer event generation configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 18 - **2D** *(W)*

Transfer type configuration bitfield:

- *0b0*: linear transfer
- *0b1*: 2D transfer

Bit 17 - **INC** *(W)*

Transfer incremental configuration bitfield:

- *0b0*: non incremental
- *0b1*: incremental

Bit 16 - **TYPE** *(W)*

Transfer direction configuration bitfield:

- *0b0*: L1 to L2
- *0b1*: L2 to L1

Bits 15:0 - **LEN** *(W)*

Transfer length in bytes configuration bitfield.

8.1.5.4.3 Cluster DMA transfer free command format. (STATUS)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TID_ALLOC															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TID_TR															

Bits 31:16 - **TID_ALLOC** *(R)*

Transfer status bitfield:

- TID_TR[i]=*0b0* means that transfer allocator with TID i-16 is free.
- TID_TR[i]=*0b1* means that transfer allocator with TID i-16 is reserved.

Bits 15:0 - **TID_TR** *(R)*

Transfer status bitfield:

TID_TR[i]=*0b1* means that transfer with TID i is active.

8.1.5.4.4 Cluster DMA transfer status format. (FREE_TID)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TID_FREE															

Bits 15:0 - **TID_FREE** *(W)*

Transfer canceller configuration bitfield. Writing a *0b1* in TID_FREE[i] will free transfer with TID i.

8.1.5.4.5 Cluster DMA L1 base address configuration format. (TCDM)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															

Bits 31:0 - **ADDR** *(W)*

Transfer L1 base address configuration bitfield.

8.1.5.4.6 Cluster DMA L2 base address configuration format. (EXT_L2)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR															

Bits 31:0 - **ADDR** *(W)*

Transfer L2 base address configuration bitfield.

8.1.5.4.7 Cluster DMA 2D transfer configuration format. (2D)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STRIDE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEN															

Bits 31:16 - **STRIDE** *(W)*

2D transfer stride value configuration bitfield.

Bits 15:0 - **LEN** *(W)*

2D transfer length value configuration bitfield.

8.2 Axi Lite Subsystem

8.2.1 LLC_CFG

None

8.2.1.1 Last Level Cache Config Registers registers

Name	Address	Size	Type	Access	Default	Description
CFG_SPM_LOW	0x10401000	32	Config	R/W	*0x0000*	The 32 least significant bits of cache way SPM configuration (1: SPM, 0: cache). Bit 0 specifies way 0, bit 1 specifies way 1, etc.
CFG_SPM_HIGH	0x10401004	32	Config	R/W	*0x0000*	The 32 most significant bits of cache way SPM configuration (1: SPM, 0: cache). Bit 32 specifies way 32, bit 33 specifies way 33, etc.
CFG_FLUSH_LOW	0x10401008	32	Config	R/W	*0x0000*	The 32 least significant bits of cache way flush - 1 indicates flush. Bit 0 specifies way 0, bit 1 specifies way 1, etc.
CFG_FLUSH_HIGH	0x1040100C	32	Config	R/W	*0x0000*	The 32 most significant bits of cache way flush - 1 indicates flush. Bit 32 specifies way 32, bit 33 specifies way 33, etc.
FLUSHED_LOW	0x10401010	32	Status	R	*0x0000*	The 32 least significant bits of cache way flush status - 1 indicates flush. Bit 0 specifies way 0, bit 1 specifies way 1, etc.
FLUSHED_HIGH	0x10401014	32	Status	R	*0x0000*	The 32 most significant bits of cache way flush status - 1 indicates flush. Bit 32 specifies way 32, bit 33 specifies way 33, etc.
BIST_OUT_LOW	0x10401018	32	Status	R	*0x0000*	The 32 least significant bits of cache way BIST MarchX results. Bit 0 specifies way 0, bit 1 specifies way 1, etc.

Name	Address	Size	Type	Access	Default	Description
BIST_OUT_HIGH	0x1040101C	32	Status	R	*0x0000*	The 32 most significant bits of cache way BIST MarchX results. Bit 32 specifies way 32, bit 33 specifies way 33, etc.
SET_ASSO_LOW	0x10401020	32	Status	R	*0x0008*	The 32 least significant bits of the register reporting LLC set associativity.
SET_ASSO_HIGH	0x10401024	32	Status	R	*0x0000*	The 32 most significant bits of the register reporting LLC set associativity.
NUM_LINES_LOW	0x10401028	32	Status	R	*0x0100*	The 32 least significant bits of the register reporting number of lines per LLC set.
NUM_LINES_HIGH	0x1040102C	32	Status	R	*0x0000*	The 32 most significant bits of the register reporting number of lines per LLC set.
NUM_BLOCKS_LOW	0x10401030	32	Status	R	*0x0008*	The 32 least significant bits of the register reporting number of blocks per LLC line.
NUM_BLOCKS_HIGH	0x10401034	32	Status	R	*0x0000*	The 32 most significant bits of the register reporting number of blocks per LLC line.
VERSION_LOW	0x10401038	32	Status	R	*0x30303031*	The 32 least significant bits of the register reporting LLC version.
VERSION_HIGH	0x1040103C	32	Status	R	*0x4C6C633A*	The 32 most significant bits of the register reporting LLC version.

Table 11. Last Level Cache Config Registers registers table

8.2.1.2 LLC_CFG registers details

8.2.1.2.1 The 32 least significant bits of cache way SPM configuration (1: SPM, 0: cache). Bit 0 specifies way 0, bit 1 specifies way 1, etc. (CFG_SPM_LOW)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CFG_SPM_LOW							

Bits 7:0 - **CFG_SPM_LOW** *(R/W)*

One-hot encoding of LLC way SPM mode. Bit 0 indicates way 0, bit 1 indicates way 1, …, bit 7 indicates way 7. If bit is set to 1, way is set to SPM; if it is set to 0, it is set as cache.

8.2.1.2.2 The 32 most significant bits of cache way SPM configuration (1: SPM, 0: cache). Bit 32 specifies way 32, bit 33 specifies way 33, etc. (CFG_SPM_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits -1:0 - **CFG_SPM_HIGH** *(R/W)*

Unused, only 8 ways in LLC.

8.2.1.2.3 The 32 least significant bits of cache way flush - 1 indicates flush. Bit 0 specifies way 0, bit 1 specifies way 1, etc. (CFG_FLUSH_LOW)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CFG_FLUSH_LOW							

Bits 7:0 - **CFG_FLUSH_LOW** *(R/W)*

One-hot encoding of LLC way flush request. Bit 0 indicates way 0, bit 1 indicates way 1, …, bit 7 indicates way 7. If bit is set to 1, way flush is requested.

8.2.1.2.4 The 32 most significant bits of cache way flush - 1 indicates flush. Bit 32 specifies way 32, bit 33 specifies way 33, etc. (CFG_FLUSH_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits -1:0 - **CFG_FLUSH_HIGH** *(R/W)*

Unused, only 8 ways in LLC.

8.2.1.2.5 The 32 least significant bits of cache way flush status - 1 indicates flush. Bit 0 specifies way 0, bit 1 specifies way 1, etc. (FLUSHED_LOW)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FLUSHED_LOW							

Bits 7:0 - **FLUSHED_LOW** *(R)*

One-hot encoding indicating LLC way flushed. Bit 0 indicates way 0, bit 1 indicates way 1, …, bit 7 indicates way 7. If bit is set to 1, way was flushed.

8.2.1.2.6 The 32 most significant bits of cache way flush status - 1 indicates flush. Bit 32 specifies way 32, bit 33 specifies way 33, etc. (FLUSHED_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits -1:0 - **FLUSHED_HIGH** *(R)*

Unused, only 8 ways in LLC.

8.2.1.2.7 The 32 least significant bits of cache way BIST MarchX results. Bit 0 specifies way 0, bit 1 specifies way 1, etc. (BIST_OUT_LOW)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BIST_OUT_LOW							

Bits 7:0 - **BIST_OUT_LOW** *(R)*

One-hot encoding of LLC way BIST MarchX results. Bit 0 indicates way 0, bit 1 indicates way 1, …, bit 7 indicates way 7.

8.2.1.2.8 The 32 most significant bits of cache way BIST MarchX results. Bit 32 specifies way 32, bit 33 specifies way 33, etc. (BIST_OUT_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits -1:0 - **BIST_OUT_HIGH** *(R)*

Unused, only 8 ways in LLC.

8.2.1.2.9 The 32 least significant bits of the register reporting LLC set associativity. (SET_ASSO_LOW)

Reset value: 0x0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SET_ASSO_LOW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET_ASSO_LOW															

Bits 31:0 - **SET_ASSO_LOW** *(R)*

Least significant bits of the status register indicating LLC set associativity.

8.2.1.2.10 The 32 most significant bits of the register reporting LLC set associativity. (SET_ASSO_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SET_ASSO_HIGH															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET_ASSO_HIGH															

Bits 31:0 - **SET_ASSO_HIGH** *(R)*

Most significant bits of the status register indicating LLC set associativity.

8.2.1.2.11 The 32 least significant bits of the register reporting number of lines per LLC set. (NUM_LINES_LOW)

Reset value: 0x0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_LINES_LOW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_LINES_LOW															

Bits 31:0 - **NUM_LINES_LOW** *(R)*

Least significant bits of the status register indicating number of lines per LLC set.

8.2.1.2.12 The 32 most significant bits of the register reporting number of lines per LLC set. (NUM_LINES_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_LINES_HIGH															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_LINES_HIGH															

Bits 31:0 - **NUM_LINES_HIGH** *(R)*

Most significant bits of the status register indicating number of lines per LLC set.

8.2.1.2.13 The 32 least significant bits of the register reporting number of blocks per LLC line. (NUM_BLOCKS_LOW)

Reset value: 0x0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_BLOCKS_LOW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_BLOCKS_LOW															

Bits 31:0 - **NUM_BLOCKS_LOW** *(R)*

Least significant bits of the status register indicating number of blocks per LLC line.

8.2.1.2.14 The 32 most significant bits of the register reporting number of blocks per LLC line. (NUM_BLOCKS_HIGH)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_BLOCKS_HIGH															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_BLOCKS_HIGH															

Bits 31:0 - **NUM_BLOCKS_HIGH** *(R)*

Most significant bits of the status register indicating number of blocks per LLC line.

8.2.1.2.15 The 32 least significant bits of the register reporting LLC version. (VERSION_LOW)

Reset value: 0x30303031

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VERSION_LOW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION_LOW															

Bits 31:0 - **VERSION_LOW** *(R)*

Least significant bits of the status register indicating LLC version.

8.2.1.2.16 The 32 most significant bits of the register reporting LLC version. (VERSION_HIGH)

Reset value: 0x4C6C633A

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VERSION_HIGH															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION_HIGH															

Bits 31:0 - **VERSION_HIGH** *(R)*

Most significant bits of the status register indicating LLC version.

8.3 System Timer

8.3.1 CVA6 Timer

None

8.3.1.1 System Timer registers

Name	Address	Size	Type	Access	Default	Description
CFG_LO	0x18000000	32	Config	R/W	*0x0000*	Timer Low Configuration register.
CFG_HI	0x18000004	32	Config	R/W	*0x0000*	Timer High Configuration register.
CNT_LO	0x18000008	32	Data	R/W	*0x0000*	Timer Low counter value register.
CNT_HI	0x1800000C	32	Data	R/W	*0x0000*	Timer High counter value register.
CMP_LO	0x18000010	32	Config	R/W	*0x0000*	Timer Low comparator value register.
CMP_HI	0x18000014	32	Config	R/W	*0x0000*	Timer High comparator value register.
START_LO	0x18000018	32	Config	R/W	*0x0000*	Start Timer Low counting register.

Name	Address	Size	Type	Access	Default	Description
START_HI	0x1800001C	32	Config	R/W	*0x0000*	Start Timer High counting register.
RESET_LO	0x18000020	32	Config	R/W	*0x0000*	Reset Timer Low counter register.
RESET_HI	0x18000024	32	Config	R/W	*0x0000*	Reset Timer High counter register.

Table 12. System Timer registers table

Registers details are identical to Basic timer defined above

8.4 APB Subsystem

8.4.1 HYPER_CFG

None

8.4.1.1 HYPERBUS_AXI registers

Name	Address	Size	Type	Access	Default	Description
T_LATENCY ACCESS	0x1A101000	32	Config	R/W	*0x0006*	Initial latency
EN_LATENCY_ADDITIONAL	0x1A101004	32	Config	R/W	*0x0000*	Force 2x Latency count
T_BURST_MAX	0x1A101008	32	Config	R/W	*0x015E*	Max burst Length between two memory refresh
T_READ_WRITE_RECOVERY	0x1A10100C	32	Config	R/W	*0x0006*	Idle time between transactions
T_RX_CLOCK_DELAY	0x1A101010	32	Config	R/W	*0x0008*	RX Delay Line
T_TX_CLOCK_DELAY	0x1A101014	32	Config	R/W	*0x0008*	TX Delay Line
ADDRESS_MASK_MSB	0x1A101018	32	Config	R/W	*0x0019*	Address Mask MSB
ADDRESS_SPACE	0x1A10101C	32	Config	R/W	*0x0000*	Select the address space between memory and config regiters
PHYS_IN_USE	0x1A101020	32	Config	R/W	*0x0001*	Number of PHYs on use
WHICH_PHY	0x1A101024	32	Config	R/W	*0x0001*	PHY used in single PHY mode
CS0_BASE	0x1A101028	32	Config	R/W	*0x80000000*	CS0 Base address range
CS0_END	0x1A10102C	32	Config	R/W	*0x81000000*	CS0 End address range
CS1_BASE	0x1A101030	32	Config	R/W	*0x81000000*	CS1 Base address range
CS1_END	0x1A101034	32	Config	R/W	*0x82000000*	CS1 End address range
CS2_BASE	0x1A101038	32	Config	R/W	*0x82000000*	CS2 Base address range
CS2_END	0x1A10103C	32	Config	R/W	*0x83000000*	CS2 End address range
CS3_BASE	0x1A101040	32	Config	R/W	*0x83000000*	CS3 Base address range
CS3_END	0x1A101044	32	Config	R/W	*0x84000000*	CS3 End address range

Table 13. HYPERBUS_AXI registers table

8.4.1.2 HYPER_CFG registers details

8.4.1.2.1 Initial latency (T_LATENCY ACCESS)

Reset value: 0x0006

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												T_LATENCY ACCESS			

Bits 3:0 - **T_LATENCY ACCESS** *(R/W)*

Reset value: 0b0110

Memory space read and write transactions or register space read transactions require some initial latency to open the row selected by the CA. This initial latency is tACC. The number of latency clocks needed to satisfy tACC depends on the HyperBus frequency can vary from 3 to 7 clocks. (Refere to the memory bank datasheet)

8.4.1.2.2 Force 2x Latency count (EN_LATENCY_ADDITIONAL)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															EN_LATENCY_ADDITIONAL

Bit 0 - **EN_LATENCY_ADDITIONAL** *(R/W)*

Enable Latency count 2x:

- 1'b0: Disabled
- 1'b1: Enabled

8.4.1.2.3 Max burst Length between two memory refresh (T_BURST_MAX)

Reset value: 0x015E

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
T_BURST_MAX															

Bits 15:0 - **T_BURST_MAX** *(R/W)*
Reset value: 0x15E

Max burst Length between two memory refresh

8.4.1.2.4 Idle time between transactions (T_READ_WRITE_RECOVERY)

Reset value: 0x0006

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												T_READ_WRITE_RECOVERY			

Bits 3:0 - **T_READ_WRITE_RECOVERY** *(R/W)*
Reset value: 0b0110

Idle time between transactions

8.4.1.2.5 RX Delay Line (T_RX_CLOCK_DELAY)

Reset value: 0x0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												T_RX_CLOCK_DELAY			

Bits 3:0 - **T_RX_CLOCK_DELAY** *(R/W)*
Reset value: 0b1000

RX Delay Line

8.4.1.2.6 TX Delay Line (T_TX_CLOCK_DELAY)

Reset value: 0x0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												T_TX_CLOCK_DELAY			

Bits 3:0 - **T_TX_CLOCK_DELAY** *(R/W)*
Reset value: 0b1000

TX Delay Line

8.4.1.2.7 Address Mask MSB (ADDRESS_MASK_MSB)

Reset value: 0x0019

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ADDRESS_MASK_MSB		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS_MASK_MSB															

Bits 18:0 - **ADDRESS_MASK_MSB** *(R/W)*
Reset value: 0x019

Address Mask MSB

8.4.1.2.8 Select the address space between memory and config regiters (ADDRESS_SPACE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ADDRESS_SPACE

Bit 0 - **ADDRESS_SPACE** *(R/W)*

Hyperram bank address space:

- 1'b0: Access to the memory bank address space
- 1'b1: Access to the memory bank config register space

8.4.1.2.9 Number of PHYs on use (PHYS_IN_USE)

Reset value: 0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PHYS_IN_USE

Bit 0 - **PHYS_IN_USE** *(R/W)*
Reset value: 0b1

Number of PHYs on use:

- 1'b0: Uses 1 PHY
- 1'b1: Uses 2 PHYs

8.4.1.2.10 PHY used in single PHY mode (WHICH_PHY)

Reset value: 0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WHICH_PHY

Bit 0 - **WHICH_PHY** *(R/W)*

Reset value: 0b1

PHY used in single PHY mode:

- 1'b0: PHY 0 is used
- 1'b1: PHY 1 is used

8.4.1.2.11 CS0 Base address range (CS0_BASE)

Reset value: 0x80000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS0_BASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS0_BASE															

Bits 31:0 - **CS0_BASE** *(R/W)*

CS0 Base address range

8.4.1.2.12 CS0 End address range (CS0_END)

Reset value: 0x81000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS0_END															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS0_END															

Bits 31:0 - **CS0_END** *(R/W)*

CS0 End address range

8.4.1.2.13 CS1 Base address range (CS1_BASE)

Reset value: 0x81000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS1_BASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS1_BASE															

Bits 31:0 - **CS1_BASE** *(R/W)*

CS1 Base address range

8.4.1.2.14 CS1 End address range (CS1_END)

Reset value: 0x82000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS1_END															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS1_END															

Bits 31:0 - **CS1_END** *(R/W)*

CS1 End address range

8.4.1.2.15 CS2 Base address range (CS2_BASE)

Reset value: 0x82000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS2_BASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS2_BASE															

Bits 31:0 - **CS2_BASE** *(R/W)*

CS2 Base address range

8.4.1.2.16 CS2 End address range (CS2_END)

Reset value: 0x83000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS2_END															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS2_END															

Bits 31:0 - **CS2_END** *(R/W)*

CS2 End address range

8.4.1.2.17 CS3 Base address range (CS3_BASE)

Reset value: 0x83000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS3_BASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS3_BASE															

Bits 31:0 - **CS3_BASE** *(R/W)*

CS3 Base address range

8.4.1.2.18 CS3 End address range (CS3_END)

Reset value: 0x84000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS3_END															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CS3_END															

Bits 31:0 - **CS3_END** *(R/W)*

CS3 End address range

8.4.2 Advanced Timerss

None

8.4.2.1 PWM0 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A103000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A103004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A103008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A10300C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A103010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A103014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A103018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A10302C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A103040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.

Name	Address	Size	Type	Access	Default	Description
T1_CONFIG	0x1A103044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A103048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A10304C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A103050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A103054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A103058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A10306C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A103080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A103084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A103088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A10308C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A103090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A103094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A103098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A1030AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A1030C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A1030C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A1030C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A1030CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A1030D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A1030D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A1030D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A1030EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A103100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A103104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 14. PWM0 registers table

8.4.2.2 PWM1 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A223000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A223004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A223008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22300C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A223010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A223014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A223018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22302C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A223040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A223044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A223048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22304C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A223050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A223054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A223058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22306C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A223080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A223084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A223088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22308C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A223090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A223094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.

Name	Address	Size	Type	Access	Default	Description
T2_TH_CHANNEL3	0x1A223098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2230AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2230C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2230C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2230C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2230CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2230D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2230D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2230D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2230EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A223100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A223104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 15. PWM1 registers table

8.4.2.3 PWM2 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A224000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A224004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A224008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22400C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A224010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A224014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A224018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22402C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A224040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A224044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A224048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22404C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A224050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A224054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A224058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22406C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A224080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A224084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A224088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22408C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A224090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A224094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A224098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2240AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2240C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2240C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2240C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2240CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2240D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2240D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2240D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2240EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A224100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A224104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 16. PWM2 registers table

Name	Address	Size	Type	Access	Default	Description

8.4.2.4 PWM3 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A225000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A225004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A225008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22500C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A225010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A225014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A225018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22502C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A225040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A225044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A225048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22504C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A225050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A225054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A225058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22506C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A225080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A225084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A225088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22508C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A225090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A225094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A225098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2250AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2250C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2250C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2250C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2250CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2250D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2250D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2250D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2250EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A225100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A225104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 17. PWM3 registers table

8.4.2.5 PWM4 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A226000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A226004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A226008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22600C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A226010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A226014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A226018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22602C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A226040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A226044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.

Name	Address	Size	Type	Access	Default	Description
T1_THRESHOLD	0x1A226048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22604C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A226050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A226054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A226058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22606C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A226080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A226084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A226088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22608C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A226090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A226094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A226098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2260AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2260C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2260C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2260C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2260CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2260D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2260D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2260D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2260EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A226100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A226104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 18. PWM4 registers table

8.4.2.6 PWM5 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A227000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A227004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A227008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22700C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A227010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A227014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A227018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22702C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A227040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A227044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A227048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22704C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A227050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A227054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A227058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22706C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A227080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A227084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A227088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22708C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A227090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A227094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A227098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.

Name	Address	Size	Type	Access	Default	Description
T2_COUNTER	0x1A2270AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2270C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2270C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2270C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2270CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2270D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2270D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2270D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2270EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A227100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A227104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 19. PWM5 registers table

8.4.2.7 PWM6 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A228000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A228004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A228008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22800C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A228010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A228014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A228018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22802C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A228040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A228044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A228048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22804C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A228050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A228054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A228058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22806C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A228080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A228084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A228088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22808C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A228090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A228094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A228098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2280AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2280C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2280C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2280C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2280CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2280D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2280D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2280D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2280EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A228100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A228104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 20. PWM6 registers table

8.4.2.8 PWM7 registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A229000	32	Config	R/W	*0x0000*	ADV_TIMER0 command register.
T0_CONFIG	0x1A229004	32	Config	R/W	*0x0000*	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A229008	32	Config	R/W	*0x0000*	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A22900C	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A229010	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A229014	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A229018	32	Config	R/W	*0x0000*	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A22902C	32	Status	R	*0x0000*	ADV_TIMER0 counter register.
T1_CMD	0x1A229040	32	Config	R/W	*0x0000*	ADV_TIMER1 command register.
T1_CONFIG	0x1A229044	32	Config	R/W	*0x0000*	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A229048	32	Config	R/W	*0x0000*	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A22904C	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A229050	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A229054	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 2 threshold configuration register.
T1_TH_CHANNEL3	0x1A229058	32	Config	R/W	*0x0000*	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A22906C	32	Status	R	*0x0000*	ADV_TIMER1 counter register.
T2_CMD	0x1A229080	32	Config	R/W	*0x0000*	ADV_TIMER2 command register.
T2_CONFIG	0x1A229084	32	Config	R/W	*0x0000*	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A229088	32	Config	R/W	*0x0000*	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A22908C	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A229090	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A229094	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A229098	32	Config	R/W	*0x0000*	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A2290AC	32	Status	R	*0x0000*	ADV_TIMER2 counter register.
T3_CMD	0x1A2290C0	32	Config	R/W	*0x0000*	ADV_TIMER3 command register.
T3_CONFIG	0x1A2290C4	32	Config	R/W	*0x0000*	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A2290C8	32	Config	R/W	*0x0000*	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A2290CC	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A2290D0	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A2290D4	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 2 threshold configuration register.
T3_TH_CHANNEL3	0x1A2290D8	32	Config	R/W	*0x0000*	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A2290EC	32	Status	R	*0x0000*	ADV_TIMER3 counter register.
EVENT_CFG	0x1A229100	32	Config	R/W	*0x0000*	ADV_TIMERS events configuration register.
CG	0x1A229104	32	Config	R/W	*0x0000*	ADV_TIMERS channels clock gating configuration register.

Table 21. PWM7 registers table

8.4.2.9 Advanced Timers registers details

8.4.2.9.1 ADV_TIMER0 command register. (T0_CMD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved											ARM	RESET	UPDATE	STOP	START

Bit 4 - **ARM** *(R/W)*

ADV_TIMER0 arm command bitfield.

Bit 3 - **RESET** *(R/W)*

ADV_TIMER0 reset command bitfield.

Bit 2 - **UPDATE** *(R/W)*

ADV_TIMER0 update command bitfield.

Bit 1 - **STOP** *(R/W)*

ADV_TIMER0 stop command bitfield.

Bit 0 - **START** *(R/W)*

ADV_TIMER0 start command bitfield.

8.4.2.9.2 ADV_TIMER0 configuration register. (T0_CONFIG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PRESC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			UPDOWNSEL	CLKSEL	MODE			INSEL							

Bits 23:16 - **PRESC** *(R/W)*

ADV_TIMER0 prescaler value configuration bitfield.

Bit 12 - **UPDOWNSEL** *(R/W)*

ADV_TIMER0 center-aligned mode configuration bitfield:

- *0b0*: The counter counts up and down alternatively.
- *0b1*: The counter counts up and resets to 0 when reach threshold.

Bit 11 - **CLKSEL** *(R/W)*

ADV_TIMER0 clock source configuration bitfield:

- *0b0*: FLL
- *0b1*: reference clock at 32kHz

Bits 10:8 - **MODE** *(R/W)*

ADV_TIMER0 trigger mode configuration bitfield:

- *0b000*: trigger event at each clock cycle.
- *0b001*: trigger event if input source is 0
- *0b010*: trigger event if input source is 1
- *0b011*: trigger event on input source rising edge
- *0b100*: trigger event on input source falling edge
- *0b101*: trigger event on input source falling or rising edge
- *0b110*: trigger event on input source rising edge when armed
- *0b111*: trigger event on input source falling edge when armed

Bits 7:0 - **INSEL** *(R/W)*

ADV_TIMER0 input source configuration bitfield:

- 0–31: GPIO[0] to GPIO[31]
- 32–35: Channel 0 to 3 of ADV_TIMER0
- 36–39: Channel 0 to 3 of ADV_TIMER1
- 40–43: Channel 0 to 3 of ADV_TIMER2
- 44–47: Channel 0 to 3 of ADV_TIMER3

8.4.2.9.3 ADV_TIMER0 threshold configuration register. (T0_THRESHOLD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TH_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH_LO															

Bits 31:16 - **TH_HI** *(R/W)*

ADV_TIMER0 threshold high part configuration bitfield. It defines end counter value.

Bits 15:0 - **TH_LO** *(R/W)*

ADV_TIMER0 threshold low part configuration bitfield. It defines start counter value.

8.4.2.9.4 ADV_TIMER0 channel 0 threshold configuration register. (T0_TH_CHANNEL0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER0 channel 0 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER0 channel 0 threshold configuration bitfield.

8.4.2.9.5 ADV_TIMER0 channel 1 threshold configuration register. (T0_TH_CHANNEL1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER0 channel 1 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER0 channel 1 threshold configuration bitfield.

8.4.2.9.6 ADV_TIMER0 channel 2 threshold configuration register. (T0_TH_CHANNEL2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER0 channel 2 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER0 channel 2 threshold configuration bitfield.

8.4.2.9.7 ADV_TIMER0 channel 3 threshold configuration register. (T0_TH_CHANNEL3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER0 channel 3 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER0 channel 3 threshold configuration bitfield.

8.4.2.9.8 ADV_TIMER1 command register. (T1_CMD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved											ARM	RESET	UPDATE	STOP	START

Bit 4 - **ARM** *(R/W)*

ADV_TIMER1 arm command bitfield.

Bit 3 - **RESET** *(R/W)*

ADV_TIMER1 reset command bitfield.

Bit 2 - **UPDATE** *(R/W)*

ADV_TIMER1 update command bitfield.

Bit 1 - **STOP** *(R/W)*

ADV_TIMER1 stop command bitfield.

Bit 0 - **START** *(R/W)*

ADV_TIMER1 start command bitfield.

8.4.2.9.9 ADV_TIMER1 configuration register. (T1_CONFIG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PRESC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			UPDOWNSEL	CLKSEL	MODE			INSEL							

Bits 23:16 - **PRESC** *(R/W)*

ADV_TIMER1 prescaler value configuration bitfield.

Bit 12 - **UPDOWNSEL** *(R/W)*

ADV_TIMER1 center-aligned mode configuration bitfield:

- *0b0*: The counter counts up and down alternatively.
- *0b1*: The counter counts up and resets to 0 when reach threshold.

Bit 11 - **CLKSEL** *(R/W)*

ADV_TIMER1 clock source configuration bitfield:

- *0b0*: FLL
- *0b1*: reference clock at 32kHz

Bits 10:8 - **MODE** *(R/W)*

ADV_TIMER1 trigger mode configuration bitfield:

- *0b000*: trigger event at each clock cycle.
- *0b001*: trigger event if input source is 0
- *0b010*: trigger event if input source is 1
- *0b011*: trigger event on input source rising edge
- *0b100*: trigger event on input source falling edge
- *0b101*: trigger event on input source falling or rising edge
- *0b110*: trigger event on input source rising edge when armed
- *0b111*: trigger event on input source falling edge when armed

Bits 7:0 - **INSEL** *(R/W)*

ADV_TIMER1 input source configuration bitfield:

- 0–31: GPIO[0] to GPIO[31]
- 32–35: Channel 0 to 3 of ADV_TIMER0
- 36–39: Channel 0 to 3 of ADV_TIMER1
- 40–43: Channel 0 to 3 of ADV_TIMER2
- 44–47: Channel 0 to 3 of ADV_TIMER3

8.4.2.9.10 ADV_TIMER1 threshold configuration register. (T1_THRESHOLD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TH_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH_LO															

Bits 31:16 - **TH_HI** *(R/W)*

ADV_TIMER1 threshold high part configuration bitfield. It defines end counter value.

Bits 15:0 - **TH_LO** *(R/W)*

ADV_TIMER1 threshold low part configuration bitfield. It defines start counter value.

8.4.2.9.11 ADV_TIMER1 channel 0 threshold configuration register. (T1_TH_CHANNEL0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER1 channel 0 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER1 channel 0 threshold configuration bitfield.

8.4.2.9.12 ADV_TIMER1 channel 1 threshold configuration register. (T1_TH_CHANNEL1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER1 channel 1 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER1 channel 1 threshold configuration bitfield.

8.4.2.9.13 ADV_TIMER1 channel 2 threshold configuration register. (T1_TH_CHANNEL2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER1 channel 2 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER1 channel 2 threshold configuration bitfield.

8.4.2.9.14 ADV_TIMER1 channel 3 threshold configuration register. (T1_TH_CHANNEL3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER1 channel 3 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER1 channel 3 threshold configuration bitfield.

8.4.2.9.15 ADV_TIMER2 command register. (T2_CMD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved											ARM	RESET	UPDATE	STOP	START

Bit 4 - **ARM** *(R/W)*

ADV_TIMER2 arm command bitfield.

Bit 3 - **RESET** *(R/W)*

ADV_TIMER2 reset command bitfield.

Bit 2 - **UPDATE** *(R/W)*

ADV_TIMER2 update command bitfield.

Bit 1 - **STOP** *(R/W)*

ADV_TIMER2 stop command bitfield.

Bit 0 - **START** *(R/W)*

ADV_TIMER2 start command bitfield.

8.4.2.9.16 ADV_TIMER2 configuration register. (T2_CONFIG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PRESC							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			UPDOWNSEL	CLKSEL	MODE			INSEL							

Bits 23:16 - **PRESC** *(R/W)*

ADV_TIMER2 prescaler value configuration bitfield.

Bit 12 - **UPDOWNSEL** *(R/W)*

ADV_TIMER2 center-aligned mode configuration bitfield:

- *0b0*: The counter counts up and down alternatively.
- *0b1*: The counter counts up and resets to 0 when reach threshold.

Bit 11 - **CLKSEL** *(R/W)*

ADV_TIMER2 clock source configuration bitfield:

- *0b0*: FLL
- *0b1*: reference clock at 32kHz

Bits 10:8 - **MODE** *(R/W)*

ADV_TIMER2 trigger mode configuration bitfield:

- *0b000*: trigger event at each clock cycle.
- *0b001*: trigger event if input source is 0
- *0b010*: trigger event if input source is 1
- *0b011*: trigger event on input source rising edge
- *0b100*: trigger event on input source falling edge
- *0b101*: trigger event on input source falling or rising edge
- *0b110*: trigger event on input source rising edge when armed
- *0b111*: trigger event on input source falling edge when armed

Bits 7:0 - **INSEL** *(R/W)*

ADV_TIMER2 input source configuration bitfield:

- 0–31: GPIO[0] to GPIO[31]
- 32–35: Channel 0 to 3 of ADV_TIMER0
- 36–39: Channel 0 to 3 of ADV_TIMER1
- 40–43: Channel 0 to 3 of ADV_TIMER2
- 44–47: Channel 0 to 3 of ADV_TIMER3

8.4.2.9.17 ADV_TIMER2 threshold configuration register. (T2_THRESHOLD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TH_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH_LO															

Bits 31:16 - **TH_HI** *(R/W)*

ADV_TIMER2 threshold high part configuration bitfield. It defines end counter value.

Bits 15:0 - **TH_LO** *(R/W)*

ADV_TIMER2 threshold low part configuration bitfield. It defines start counter value.

8.4.2.9.18 ADV_TIMER2 channel 0 threshold configuration register. (T2_TH_CHANNEL0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER2 channel 0 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER2 channel 0 threshold configuration bitfield.

8.4.2.9.19 ADV_TIMER2 channel 1 threshold configuration register. (T2_TH_CHANNEL1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER2 channel 1 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER2 channel 1 threshold configuration bitfield.

8.4.2.9.20 ADV_TIMER2 channel 2 threshold configuration register. (T2_TH_CHANNEL2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER2 channel 2 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER2 channel 2 threshold configuration bitfield.

8.4.2.9.21 ADV_TIMER2 channel 3 threshold configuration register. (T2_TH_CHANNEL3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER2 channel 3 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER2 channel 3 threshold configuration bitfield.

8.4.2.9.22 ADV_TIMER3 command register. (T3_CMD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved											ARM	RESET	UPDATE	STOP	START

Bit 4 - **ARM** *(R/W)*

ADV_TIMER3 arm command bitfield.

Bit 3 - **RESET** *(R/W)*

ADV_TIMER3 reset command bitfield.

Bit 2 - **UPDATE** *(R/W)*

ADV_TIMER3 update command bitfield.

Bit 1 - **STOP** *(R/W)*

ADV_TIMER3 stop command bitfield.

Bit 0 - **START** *(R/W)*

ADV_TIMER3 start command bitfield.

8.4.2.9.23 ADV_TIMER3 configuration register. (T3_CONFIG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PRESC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			UPDOWNSEL	CLKSEL	MODE			INSEL							

Bits 23:16 - **PRESC** *(R/W)*

ADV_TIMER3 prescaler value configuration bitfield.

Bit 12 - **UPDOWNSEL** *(R/W)*

ADV_TIMER3 center-aligned mode configuration bitfield:

- *0b0*: The counter counts up and down alternatively.
- *0b1*: The counter counts up and resets to 0 when reach threshold.

Bit 11 - **CLKSEL** *(R/W)*

ADV_TIMER3 clock source configuration bitfield:

- *0b0*: FLL
- *0b1*: reference clock at 32kHz

Bits 10:8 - **MODE** *(R/W)*

ADV_TIMER3 trigger mode configuration bitfield:

- *0b000*: trigger event at each clock cycle.
- *0b001*: trigger event if input source is 0
- *0b010*: trigger event if input source is 1
- *0b011*: trigger event on input source rising edge
- *0b100*: trigger event on input source falling edge
- *0b101*: trigger event on input source falling or rising edge
- *0b110*: trigger event on input source rising edge when armed
- *0b111*: trigger event on input source falling edge when armed

Bits 7:0 - **INSEL** *(R/W)*

ADV_TIMER3 input source configuration bitfield:

- 0–31: GPIO[0] to GPIO[31]
- 32–35: Channel 0 to 3 of ADV_TIMER0
- 36–39: Channel 0 to 3 of ADV_TIMER1
- 40–43: Channel 0 to 3 of ADV_TIMER2
- 44–47: Channel 0 to 3 of ADV_TIMER3

8.4.2.9.24 ADV_TIMER3 threshold configuration register. (T3_THRESHOLD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TH_HI															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH_LO															

Bits 31:16 - **TH_HI** *(R/W)*

ADV_TIMER3 threshold high part configuration bitfield. It defines end counter value.

Bits 15:0 - **TH_LO** *(R/W)*

ADV_TIMER3 threshold low part configuration bitfield. It defines start counter value.

8.4.2.9.25 ADV_TIMER3 channel 0 threshold configuration register. (T3_TH_CHANNEL0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER3 channel 0 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER3 channel 0 threshold configuration bitfield.

8.4.2.9.26 ADV_TIMER3 channel 1 threshold configuration register. (T3_TH_CHANNEL1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER3 channel 1 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER3 channel 1 threshold configuration bitfield.

8.4.2.9.27 ADV_TIMER3 channel 2 threshold configuration register. (T3_TH_CHANNEL2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER3 channel 2 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER3 channel 2 threshold configuration bitfield.

8.4.2.9.28 ADV_TIMER3 channel 3 threshold configuration register. (T3_TH_CHANNEL3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													MODE		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TH															

Bits 18:16 - **MODE** *(R/W)*

ADV_TIMER3 channel 3 threshold match action on channel output signal configuration bitfield:

- *0b000*: set.
- *0b001*: toggle then next threshold match action is clear.
- *0b010*: set then next threshold match action is clear.
- *0b011*: toggle.
- *0b100*: clear.
- *0b101*: toggle then next threshold match action is set.
- *0b110*: clear then next threshold match action is set.

Bits 15:0 - **TH** *(R/W)*

ADV_TIMER3 channel 3 threshold configuration bitfield.

8.4.2.9.29 ADV_TIMERS events configuration register. (EVENT_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												ENA			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEL3				SEL2				SEL1				SEL0			

Bits 19:16 - **ENA** *(R/W)*

ADV_TIMER output event enable configuration bitfield. ENA[i]=1 enables output event i generation.

Bits 15:12 - **SEL3** *(R/W)*

ADV_TIMER output event 3 source configuration bitfiled:

- *0b0000*: ADV_TIMER0 channel 0.
- *0b0001*: ADV_TIMER0 channel 1.
- *0b0010*: ADV_TIMER0 channel 2.
- *0b0011*: ADV_TIMER0 channel 3.
- *0b0100*: ADV_TIMER1 channel 0.
- *0b0101*: ADV_TIMER1 channel 1.
- *0b0110*: ADV_TIMER1 channel 2.
- *0b0111*: ADV_TIMER1 channel 3.
- *0b1000*: ADV_TIMER2 channel 0.
- *0b1001*: ADV_TIMER2 channel 1.
- *0b1010*: ADV_TIMER2 channel 2.
- *0b1011*: ADV_TIMER2 channel 3.
- *0b1100*: ADV_TIMER3 channel 0.
- *0b1101*: ADV_TIMER3 channel 1.
- *0b1110*: ADV_TIMER3 channel 2.
- *0b1111*: ADV_TIMER3 channel 3.

Bits 11:8 - **SEL2** *(R/W)*

ADV_TIMER output event 2 source configuration bitfiled:

- *0b0000*: ADV_TIMER0 channel 0.
- *0b0001*: ADV_TIMER0 channel 1.
- *0b0010*: ADV_TIMER0 channel 2.
- *0b0011*: ADV_TIMER0 channel 3.
- *0b0100*: ADV_TIMER1 channel 0.
- *0b0101*: ADV_TIMER1 channel 1.
- *0b0110*: ADV_TIMER1 channel 2.
- *0b0111*: ADV_TIMER1 channel 3.
- *0b1000*: ADV_TIMER2 channel 0.
- *0b1001*: ADV_TIMER2 channel 1.
- *0b1010*: ADV_TIMER2 channel 2.
- *0b1011*: ADV_TIMER2 channel 3.
- *0b1100*: ADV_TIMER3 channel 0.
- *0b1101*: ADV_TIMER3 channel 1.
- *0b1110*: ADV_TIMER3 channel 2.
- *0b1111*: ADV_TIMER3 channel 3.

Bits 7:4 - **SEL1** *(R/W)*

ADV_TIMER output event 1 source configuration bitfiled:

- *0b0000*: ADV_TIMER0 channel 0.
- *0b0001*: ADV_TIMER0 channel 1.
- *0b0010*: ADV_TIMER0 channel 2.
- *0b0011*: ADV_TIMER0 channel 3.
- *0b0100*: ADV_TIMER1 channel 0.
- *0b0101*: ADV_TIMER1 channel 1.
- *0b0110*: ADV_TIMER1 channel 2.
- *0b0111*: ADV_TIMER1 channel 3.
- *0b1000*: ADV_TIMER2 channel 0.
- *0b1001*: ADV_TIMER2 channel 1.
- *0b1010*: ADV_TIMER2 channel 2.
- *0b1011*: ADV_TIMER2 channel 3.
- *0b1100*: ADV_TIMER3 channel 0.
- *0b1101*: ADV_TIMER3 channel 1.
- *0b1110*: ADV_TIMER3 channel 2.
- *0b1111*: ADV_TIMER3 channel 3.

Bits 3:0 - **SEL0** *(R/W)*

ADV_TIMER output event 0 source configuration bitfiled:

- *0b0000*: ADV_TIMER0 channel 0.
- *0b0001*: ADV_TIMER0 channel 1.
- *0b0010*: ADV_TIMER0 channel 2.
- *0b0011*: ADV_TIMER0 channel 3.
- *0b0100*: ADV_TIMER1 channel 0.
- *0b0101*: ADV_TIMER1 channel 1.
- *0b0110*: ADV_TIMER1 channel 2.
- *0b0111*: ADV_TIMER1 channel 3.
- *0b1000*: ADV_TIMER2 channel 0.
- *0b1001*: ADV_TIMER2 channel 1.
- *0b1010*: ADV_TIMER2 channel 2.
- *0b1011*: ADV_TIMER2 channel 3.
- *0b1100*: ADV_TIMER3 channel 0.
- *0b1101*: ADV_TIMER3 channel 1.
- *0b1110*: ADV_TIMER3 channel 2.
- *0b1111*: ADV_TIMER3 channel 3.

8.4.2.9.30 ADV_TIMERS channels clock gating configuration register. (CG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENA															

Bits 15:0 - **ENA** *(R/W)*

ADV_TIMER clock gating configuration bitfield.

- ENA[i]=0: clock gate ADV_TIMERi.
- ENA[i]=1: enable ADV_TIMERi.

8.4.3 APB GPIO

None

8.4.3.1 GPIO Config Registers registers

Name	Address	Size	Type	Access	Default	Description
PADDIR	0x1A105000	32	Config	R/W	*0x0000*	GPIO pad direction configuration register.
PADIN	0x1A105004	32	Config	R	*0x0000*	GPIO pad input value register.
PADOUT	0x1A105008	32	Config	R/W	*0x0000*	GPIO pad output value register.
INTEN	0x1A10500C	32	Config	R/W	*0x0000*	GPIO pad interrupt enable configuration register.
INTTYPE0	0x1A105010	32	Config	R/W	*0x0000*	GPIO pad interrupt type bit 0 configuration register.
INTTYPE1	0x1A105014	32	Config	R/W	*0x0000*	GPIO pad interrupt type bit 1 configuration register.
INTSTATUS	0x1A105018	32	Status	R	*0x0000*	GPIO pad interrupt status register.
GPIOEN	0x1A10501C	32	Config	R/W	*0x0000*	GPIO pad enable configuration register.
PADCFG0	0x1A105020	32	Config	R/W	*0x0000*	GPIO pad pin 0 to 3 configuration register.
PADCFG1	0x1A105024	32	Config	R/W	*0x0000*	GPIO pad pin 4 to 7 configuration register.
PADCFG2	0x1A105028	32	Config	R/W	*0x0000*	GPIO pad pin 8 to 11 configuration register.
PADCFG3	0x1A10502C	32	Config	R/W	*0x0000*	GPIO pad pin 12 to 15 configuration register.
PADCFG4	0x1A105030	32	Config	R/W	*0x0000*	GPIO pad pin 16 to 19 configuration register.
PADCFG5	0x1A105034	32	Config	R/W	*0x0000*	GPIO pad pin 20 to 23 configuration register.
PADCFG6	0x1A105038	32	Config	R/W	*0x0000*	GPIO pad pin 24 to 27 configuration register.
PADCFG7	0x1A10503C	32	Config	R/W	*0x0000*	GPIO pad pin 28 to 31 configuration register.

Table 22. GPIO Config Registers registers table

8.4.3.2 APB GPIO registers details

8.4.3.2.1 GPIO pad direction configuration register. (PADDIR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR															

Bits 31:0 - **DIR** *(R/W)*

GPIO direction configuration bitfield:

- bit[i]=*0b0*: Input mode for GPIO[i]
- bit[i]=*0b1*: Output mode for GPIO[i]

8.4.3.2.2 GPIO pad input value register. (PADIN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA_IN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_IN															

Bits 31:0 - **DATA_IN** *(R)*

GPIO input data read bitfield. DATA_IN[i] corresponds to input data of GPIO[i].

8.4.3.2.3 GPIO pad output value register. (PADOUT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA_OUT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_OUT															

Bits 31:0 - **DATA_OUT** *(R/W)*

GPIO output data read bitfield. DATA_OUT[i] corresponds to output data set on GPIO[i].

8.4.3.2.4 GPIO pad interrupt enable configuration register. (INTEN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTEN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN															

Bits 31:0 - **INTEN** *(R/W)*

GPIO interrupt enable configuration bitfield:

- bit[i]=*0b0*: disable interrupt for GPIO[i]
- bit[i]=*0b1*: enable interrupt for GPIO[i]

8.4.3.2.5 GPIO pad interrupt type bit 0 configuration register. (INTTYPE0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTTYPE0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTTYPE0															

Bits 31:0 - **INTTYPE0** *(R/W)*

GPIO[15:0] interrupt type configuration bitfield:

- bit[2*i+1:2i*]=*0b00*: interrupt on falling edge for GPIO[i]
- bit[2*i+1:2i*]=*0b01*: interrupt on rising edge for GPIO[i]
- bit[2*i+1:2i*]=*0b10*: interrupt on rising and falling edge for GPIO[i]
- bit[2*i+1:2i*]=*0b11*: RFU

8.4.3.2.6 GPIO pad interrupt type bit 1 configuration register. (INTTYPE1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTTYPE1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTTYPE1															

Bits 31:0 - **INTTYPE1** *(R/W)*

GPIO[31:16] interrupt type configuration bitfield:

- bit[2*i+1:2i*]=*0b00*: interrupt on falling edge for GPIO[16+i]
- bit[2*i+1:2i*]=*0b01*: interrupt on rising edge for GPIO[16+i]
- bit[2*i+1:2i*]=*0b10*: interrupt on rising and falling edge for GPIO[16+i]
- bit[2*i+1:2i*]=*0b11*: RFU

8.4.3.2.7 GPIO pad interrupt status register. (INTSTATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTSTATUS															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSTATUS															

Bits 31:0 - **INTSTATUS** *(R)*

GPIO Interrupt status flags bitfield. INTSTATUS[i]=1 when interrupt received on GPIO[i]. INTSTATUS is cleared when it is red. GPIO interrupt line is also cleared when INTSTATUS register is red.

8.4.3.2.8 GPIO pad enable configuration register. (GPIOEN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIOEN															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPIOEN															

Bits 31:0 - **GPIOEN** *(R/W)*

GPIO clock enable configuration bitfield:

- bit[i]=*0b0*: disable clock for GPIO[i]
- bit[i]=*0b1*: enable clock for GPIO[i]

GPIOs are gathered by groups of 4. The clock gating of one group is done only if all 4 GPIOs are disabled.

Clock must be enabled for a GPIO if it's direction is configured in input mode.

8.4.3.2.9 GPIO pad pin 0 to 3 configuration register. (PADCFG0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO3_DS	GPIO3_PE	Reserved						GPIO2_DS	GPIO2_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO1_DS	GPIO1_PE	Reserved						GPIO0_DS	GPIO0_PE

Bit 25 - **GPIO3_DS** *(R/W)*

GPIO[3] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO3_PE** *(R/W)*

GPIO[3] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO2_DS** *(R/W)*

GPIO[2] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO2_PE** *(R/W)*

GPIO[2] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO1_DS** *(R/W)*

GPIO[1] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO1_PE** *(R/W)*

GPIO[1] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO0_DS** *(R/W)*

GPIO[0] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO0_PE** *(R/W)*

GPIO[0] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.10 GPIO pad pin 4 to 7 configuration register. (PADCFG1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO7_DS	GPIO7_PE	Reserved						GPIO6_DS	GPIO6_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO5_DS	GPIO5_PE	Reserved						GPIO4_DS	GPIO4_PE

Bit 25 - **GPIO7_DS** *(R/W)*

GPIO[7] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO7_PE** *(R/W)*

GPIO[7] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO6_DS** *(R/W)*

GPIO[6] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO6_PE** *(R/W)*

GPIO[6] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO5_DS** *(R/W)*

GPIO[5] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO5_PE** *(R/W)*

GPIO[5] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO4_DS** *(R/W)*

GPIO[4] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO4_PE** *(R/W)*

GPIO[4] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.11 GPIO pad pin 8 to 11 configuration register. (PADCFG2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO11_DS	GPIO11_PE	Reserved						GPIO10_DS	GPIO10_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO9_DS	GPIO9_PE	Reserved						GPIO8_DS	GPIO8_PE

Bit 25 - **GPIO11_DS** *(R/W)*

GPIO[11] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO11_PE** *(R/W)*

GPIO[11] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO10_DS** *(R/W)*

GPIO[10] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO10_PE** *(R/W)*

GPIO[10] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO9_DS** *(R/W)*

GPIO[9] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO9_PE** *(R/W)*

GPIO[9] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO8_DS** *(R/W)*

GPIO[8] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO8_PE** *(R/W)*

GPIO[8] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.12 GPIO pad pin 12 to 15 configuration register. (PADCFG3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO15_DS	GPIO15_PE	Reserved						GPIO14_DS	GPIO14_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO13_DS	GPIO13_PE	Reserved						GPIO12_DS	GPIO12_PE

Bit 25 - **GPIO15_DS** *(R/W)*

GPIO[15] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO15_PE** *(R/W)*

GPIO[15] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO14_DS** *(R/W)*

GPIO[14] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO14_PE** *(R/W)*

GPIO[14] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO13_DS** *(R/W)*

GPIO[13] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO13_PE** *(R/W)*

GPIO[13] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO12_DS** *(R/W)*

GPIO[12] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO12_PE** *(R/W)*

GPIO[12] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.13 GPIO pad pin 16 to 19 configuration register. (PADCFG4)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO19_DS	GPIO19_PE	Reserved						GPIO18_DS	GPIO18_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO17_DS	GPIO17_PE	Reserved						GPIO16_DS	GPIO16_PE

Bit 25 - **GPIO19_DS** *(R/W)*

GPIO[19] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO19_PE** *(R/W)*

GPIO[19] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO18_DS** *(R/W)*

GPIO[18] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO18_PE** *(R/W)*

GPIO[18] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO17_DS** *(R/W)*

GPIO[17] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO17_PE** *(R/W)*

GPIO[17] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO16_DS** *(R/W)*

GPIO[16] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO16_PE** *(R/W)*

GPIO[16] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.14 GPIO pad pin 20 to 23 configuration register. (PADCFG5)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO23_DS	GPIO23_PE	Reserved						GPIO22_DS	GPIO22_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO21_DS	GPIO21_PE	Reserved						GPIO20_DS	GPIO20_PE

Bit 25 - **GPIO23_DS** *(R/W)*

GPIO[23] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO23_PE** *(R/W)*

GPIO[23] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO22_DS** *(R/W)*

GPIO[22] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO22_PE** *(R/W)*

GPIO[22] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO21_DS** *(R/W)*

GPIO[21] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO21_PE** *(R/W)*

GPIO[21] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO20_DS** *(R/W)*

GPIO[20] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO20_PE** *(R/W)*

GPIO[20] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.15 GPIO pad pin 24 to 27 configuration register. (PADCFG6)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO27_DS	GPIO27_PE	Reserved						GPIO26_DS	GPIO26_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO25_DS	GPIO25_PE	Reserved						GPIO24_DS	GPIO24_PE

Bit 25 - **GPIO27_DS** *(R/W)*

GPIO[27] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO27_PE** *(R/W)*

GPIO[27] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO26_DS** *(R/W)*

GPIO[26] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO26_PE** *(R/W)*

GPIO[26] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO25_DS** *(R/W)*

GPIO[25] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO25_PE** *(R/W)*

GPIO[25] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO24_DS** *(R/W)*

GPIO[24] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO24_PE** *(R/W)*

GPIO[24] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.3.2.16 GPIO pad pin 28 to 31 configuration register. (PADCFG7)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GPIO31_DS	GPIO31_PE	Reserved						GPIO30_DS	GPIO30_PE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						GPIO29_DS	GPIO29_PE	Reserved						GPIO28_DS	GPIO28_PE

Bit 25 - **GPIO31_DS** *(R/W)*

GPIO[31] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 24 - **GPIO31_PE** *(R/W)*

GPIO[31] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 17 - **GPIO30_DS** *(R/W)*

GPIO[30] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 16 - **GPIO30_PE** *(R/W)*

GPIO[30] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 9 - **GPIO29_DS** *(R/W)*

GPIO[29] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 8 - **GPIO29_PE** *(R/W)*

GPIO[29] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

Bit 1 - **GPIO28_DS** *(R/W)*

GPIO[28] drive strength configuration bitfield:

- *0b0*: low drive strength
- *0b1*: high drive strength

Bit 0 - **GPIO28_PE** *(R/W)*

GPIO[28] pull activation configuration bitfield:

- *0b0*: pull disabled
- *0b1*: pull enabled

8.4.4 SoC control unit

None

8.4.4.1 SoC Control Registers registers

Name	Address	Size	Type	Access	Default	Description
CONTROL_CLUSTER	0x1A106000	32	Config	R/W	*0x0001*	Cluster Configuration
ENABLE_LLC_COUNTERS	0x1A106004	32	Config	R/W	*0x0000*	Enable LLC performance counters
LLC_READ_MISS_CACHE	0x1A106008	32	Status	R	*0x0000*	LLC read miss counter cached accesses
LLC_READ_HIT_CACHE	0x1A10600C	32	Status	R	*0x0000*	LLC read hit counter cached accesses
LLC_WRITE_MISS_CACHE	0x1A106010	32	Status	R	*0x0000*	LLC write miss counter cached accesses
LLC_WRITE_HIT_CACHE	0x1A106014	32	Status	R	*0x0000*	LLC write hit counter cached accesses
LLC_CACHE_ADDR_START	0x1A106018	32	Config	R/W	*0x80000000*	Start address of the LLC cache region
LLC_CACHE_ADDR_END	0x1A10601C	32	Config	R/W	*0x80800000*	End address of the LLC cache region
LLC_SPM_ADDR_START	0x1A106020	32	Config	R/W	*0x70000000*	Start address of the LLC SPM region
OT_CLK_SEL	0x1A106024	32	Config	R/W	*0x0001*	Mux selector for Opentitan clock
OT_CLK_DIV	0x1A106028	32	Config	R/W	*0x0001*	Clock divider for Opentitan
OT_CLK_GATE_EN	0x1A10602C	32	Config	R/W	*0x0000*	Clock gate enable for Opentitan (active high)

Table 23. SoC Control Registers registers table

8.4.4.2 SoC control unit registers details

8.4.4.2.1 Cluster Configuration (CONTROL_CLUSTER)

Reset value: 0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved													FETCH_EN	EN_BOOT	RESET_N

Bit 2 - **FETCH_EN** *(R/W)*
Reset value: 0b0

Cluster's fetch enable (Active High):

0b0: Disabled

0b1: Enabled

Bit 1 - **EN_BOOT** *(R/W)*
Reset value: 0b0

Enable for cluster's boot (Active High):

0b0: Disabled

0b1: Enabled

Bit 0 - **RESET_N** *(R/W)*
Reset value: 0b1

Software Reset for the cluster (Active Low):

0b0: Enabled

0b1: Disabled

8.4.4.2.2 Enable LLC performance counters (ENABLE_LLC_COUNTERS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															ENABLE_LLC_COUNTERS

Bit 0 - **ENABLE_LLC_COUNTERS** *(R/W)*

LLC counters enable (Active High):

0b0: Disabled

0b1: Enabled

8.4.4.2.3 LLC read miss counter cached accesses (LLC_READ_MISS_CACHE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_READ_MISS_CACHE															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LLC_READ_MISS_CACHE															

Bits 31:0 - **LLC_READ_MISS_CACHE** *(R)*

Number of hit read

8.4.4.2.4 LLC read hit counter cached accesses (LLC_READ_HIT_CACHE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_READ_HIT_CACHE															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LLC_READ_HIT_CACHE															

Bits 31:0 - **LLC_READ_HIT_CACHE** *(R)*

Number of miss read

8.4.4.2.5 LLC write miss counter cached accesses (LLC_WRITE_MISS_CACHE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_WRITE_MISS_CACHE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLC_WRITE_MISS_CACHE															

Bits 31:0 - **LLC_WRITE_MISS_CACHE** *(R/W)*

Number of miss write

8.4.4.2.6 LLC write hit counter cached accesses (LLC_WRITE_HIT_CACHE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_WRITE_HIT_CACHE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLC_WRITE_HIT_CACHE															

Bits 31:0 - **LLC_WRITE_HIT_CACHE** *(R/W)*

Number of hit write

8.4.4.2.7 Start address of the LLC cache region (LLC_CACHE_ADDR_START)

Reset value: 0x80000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_CACHE_ADDR_START															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLC_CACHE_ADDR_START															

Bits 31:0 - **LLC_CACHE_ADDR_START** *(R/W)*

Start address of the LLC cache region

8.4.4.2.8 End address of the LLC cache region (LLC_CACHE_ADDR_END)

Reset value: 0x80800000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_CACHE_ADDR_END															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLC_CACHE_ADDR_END															

Bits 31:0 - **LLC_CACHE_ADDR_END** *(R/W)*

End address of the LLC cache region

8.4.4.2.9 Start address of the LLC SPM region (LLC_SPM_ADDR_START)

Reset value: 0x70000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLC_SPM_ADDR_START															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLC_SPM_ADDR_START															

Bits 31:0 - **LLC_SPM_ADDR_START** *(R/W)*

Start address of the LLC SPM region

8.4.4.2.10 Mux selector for Opentitan clock (OT_CLK_SEL)

Reset value: 0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														OT_CLK_SEL	

Bits 1:0 - **OT_CLK_SEL** *(R/W)*

Reset value: 0b01

LLC counters enable (Active High):

0b00: CVA6 clk

0b01: SoC clk

0b10: Cluster clk

0b11: Peripheral clk

8.4.4.2.11 Clock divider for Opentitan (OT_CLK_DIV)

Reset value: 0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	OT_CLK_DIV														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OT_CLK_DIV															

Bits 30:0 - **OT_CLK_DIV** *(R/W)*

Reset value: 0x0001

Clock divider for Opentitan

8.4.4.2.12 Clock gate enable for Opentitan (active high) (OT_CLK_GATE_EN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															OT_CLK_GATE_EN

Bit 0 - **OT_CLK_GATE_EN** *(R/W)*

Clock gate enable for Opentitan (active high):

0b0: Disabled

0b1: Enabled

8.4.5 MicroDMA Subsystem

8.4.5.1 uDMA control unit

None

8.4.5.1.1 UDMA control registers

Name	Address	Size	Type	Access	Default	Description
CFG_CG	0x1A200000	32	Config	R/W	*0x0000*	uDMA interfaces clock gate configuration register.
CFG_EVENT	0x1A200004	32	Config	R/W	*0x0000*	uDMA interfaces trigger events configuration register.

Table 24. UDMA control registers table

Name	Address	Size	Type	Access	Default	Description

8.4.5.1.2 uDMA control unit registers details

8.4.5.1.2.1 uDMA interfaces clock gate configuration register. (CFG_CG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						CPI	I2S	TCDM	I2C1	I2C0	UART	HYPER	SPIM1	SPIM0	LVDS

Bit 9 - **CPI** *(R/W)*

uDMA interfaces clock gate configuration for CPI:

- *0b0*: CPI interface clock gate is enabled
- *0b1*: CPI interface clock gate is disabled

Bit 8 - **I2S** *(R/W)*

uDMA interfaces clock gate configuration for I2S:

- *0b0*: I2S interface clock gate is enabled
- *0b1*: I2S interface clock gate is disabled

Bit 7 - **TCDM** *(R/W)*

uDMA interfaces clock gate configuration for TCDM:

- *0b0*: TCDM interface clock gate is enabled
- *0b1*: TCDM interface clock gate is disabled

Bit 6 - **I2C1** *(R/W)*

uDMA interfaces clock gate configuration for I2C1:

- *0b0*: I2C1 interface clock gate is enabled
- *0b1*: I2C1 interface clock gate is disabled

Bit 5 - **I2C0** *(R/W)*

uDMA interfaces clock gate configuration for I2C0:

- *0b0*: I2C0 interface clock gate is enabled
- *0b1*: I2C0 interface clock gate is disabled

Bit 4 - **UART** *(R/W)*

uDMA interfaces clock gate configuration for UART:

- *0b0*: UART interface clock gate is enabled
- *0b1*: UART interface clock gate is disabled

Bit 3 - **HYPER** *(R/W)*

uDMA interfaces clock gate configuration for HYPER:

- *0b0*: HYPER interface clock gate is enabled
- *0b1*: HYPER interface clock gate is disabled

Bit 2 - **SPIM1** *(R/W)*

uDMA interfaces clock gate configuration for SPIM1:

- *0b0*: SPIM1 interface clock gate is enabled
- *0b1*: SPIM1 interface clock gate is disabled

Bit 1 - **SPIM0** *(R/W)*

uDMA interfaces clock gate configuration for SPIM0:

- *0b0*: SPIM0 interface clock gate is enabled
- *0b1*: SPIM0 interface clock gate is disabled

Bit 0 - **LVDS** *(R/W)*

uDMA interfaces clock gate configuration for LVDS:

- *0b0*: LVDS interface clock gate is enabled
- *0b1*: LVDS interface clock gate is disabled

8.4.5.1.2.2 uDMA interfaces trigger events configuration register. (CFG_EVENT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVT3								EVT2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EVT1								EVT0							

Bits 31:24 - **EVT3** *(R/W)*

uDMA interfaces trigger event 3 configuration bitfield. CFG_EVT3 selects which SoC event is propageted to uDMA interface trigger event 3.

Bits 23:16 - **EVT2** *(R/W)*

uDMA interfaces trigger event 2 configuration bitfield. CFG_EVT2 selects which SoC event is propageted to uDMA interface trigger event 2.

Bits 15:8 - **EVT1** *(R/W)*

uDMA interfaces trigger event 1 configuration bitfield. CFG_EVT1 selects which SoC event is propageted to uDMA interface trigger event 1.

Bits 7:0 - **EVT0** *(R/W)*

uDMA interfaces trigger event 0 configuration bitfield. CFG_EVT0 selects which SoC event is propageted to uDMA interface trigger event 0.

8.4.5.2 uDMA UART interfaces

None

8.4.5.2.1 UART Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A201000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A201004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A201008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.
TX_SADDR	0x1A201010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A201014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A201018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.

Name	Address	Size	Type	Access	Default	Description
STATUS	0x1A201020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A201024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A201028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20102C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A201030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A201034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 25. UART Channel 0 registers table

8.4.5.2.2 UART Channel 1 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A202000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A202004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A202008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.
TX_SADDR	0x1A202010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A202014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A202018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A202020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A202024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A202028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20202C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A202030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A202034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 26. UART Channel 1 registers table

8.4.5.2.3 UART Channel 2 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A203000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A203004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A203008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.
TX_SADDR	0x1A203010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A203014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A203018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A203020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A203024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A203028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20302C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A203030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A203034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 27. UART Channel 2 registers table

8.4.5.2.4 uDMA UART interface registers details

8.4.5.2.4.1 uDMA RX UART buffer base address configuration register. (RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 15:0 - **RX_SADDR** *(R/W)*

RX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

8.4.5.2.4.2 uDMA RX UART buffer size configuration register. (RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 16:0 - **RX_SIZE** *(R/W)*

RX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.2.4.3 uDMA RX UART stream configuration register. (RX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

RX channel clear and stop transfer:

- *0b0*: disable
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

RX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

RX channel enable and start transfer bitfield:

- *0b0*: disable
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

RX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.2.4.4 uDMA TX UART buffer base address configuration register. (TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 15:0 - **TX_SADDR** *(R/W)*

TX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

8.4.5.2.4.5 uDMA TX UART buffer size configuration register. (TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															TX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 16:0 - **TX_SIZE** *(R/W)*

TX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.2.4.6 uDMA TX UART stream configuration register. (TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

TX channel clear and stop transfer bitfield:

- *0b0*: disabled
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

TX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

TX channel enable and start transfer bitfield:

- *0b0*: disabled
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

TX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.2.4.7 uDMA UART status register. (STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														RX_BUSY	TX_BUSY

Bit 1 - **RX_BUSY** *(R)*

RX busy status flag:

- *0b0*: no RX transfer on-going
- *0b1*: RX transfer on-going

Bit 0 - **TX_BUSY** *(R)*

TX busy status flag:

- *0b0*: no TX transfer on-going
- *0b1*: TX transfer on-going

8.4.5.2.4.8 UDMA UART configuration register. (SETUP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKDIV															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						RX_ENA	TX_ENA	Reserved		CLEAN_FIFO	POLLING_EN	STOP_BITS	BIT_LENGTH		PARITY_ENA

Bits 31:16 - **CLKDIV** *(R/W)*

UART Clock divider configuration bitfield. The baudrate is equal to SOC_FREQ/CLKDIV.

Bit 9 - **RX_ENA** *(R/W)*

RX transceiver configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 8 - **TX_ENA** *(R/W)*

TX transceiver configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 5 - **CLEAN_FIFO** *(R/W)*

In all mode clean the RX fifo, set 1 then set 0 to realize a reset fifo:

- *0b0*: Stop Clean the RX FIFO.
- *0b1*: Clean the RX FIFO.

Bit 4 - **POLLING_EN** *(R/W)*

When in uart read, use polling method to read the data, read interrupt enable flag will be ignored:

- *0b0*: Do not using polling method to read data.
- *0b1*: Using polling method to read data. Interrupt enable flag will be ignored.

Bit 3 - **STOP_BITS** *(R/W)*

Stop bits length bitfield:

- *0b0*: 1 stop bit
- *0b1*: 2 stop bits

Bits 2:1 - **BIT_LENGTH** *(R/W)*

Character length bitfield:

- *0b00*: 5 bits
- *0b01*: 6 bits
- *0b10*: 7 bits
- *0b11*: 8 bits

Bit 0 - **PARITY_ENA** *(R/W)*

Parity bit generation and check configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.4.5.2.4.9 uDMA UART Error status (ERROR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														RX_ERR_PARITY	RX_ERR_OVERFLOW

Bit 1 - **RX_ERR_PARITY** *(R)*

RX parity error status flag:

- *0b0*: no error
- *0b1*: RX parity error occurred

Bit 0 - **RX_ERR_OVERFLOW** *(R)*

RX overflow error status flag:

- *0b0*: no error
- *0b1*: RX overflow error occurred

8.4.5.2.4.10 uDMA UART Read or Error interrupt enable register. (IRQ_EN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ERROR	RX

Bit 1 - **ERROR** *(R/W)*

Error interrupt in enable flag:

- *0b0*: Error IRQ disable
- *0b1*: Error IRQ enable

Bit 0 - **RX** *(R/W)*

Rx interrupt in enable flag:

- *0b0*: RX IRQ disable
- *0b1*: RX IRQ enable

8.4.5.2.4.11 uDMA UART Read polling data valid flag register. (VALID)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															READY

Bit 0 - **READY** *(R)*

Used only in RX polling method to indicate data is ready for read:

- *0b0*: Data is not ready to read
- *0b1*: Data is ready to read

8.4.5.2.4.12 uDMA UART Read polling data register. (DATA)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BYTE							

Bits 7:0 - **BYTE** *(R)*

RX read data for polling or interrupt

8.4.5.3 uDMA USART interfaces

None

8.4.5.3.1 USART Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A204000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A204004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A204008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.

Name	Address	Size	Type	Access	Default	Description
TX_SADDR	0x1A204010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A204014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A204018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A204020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A204024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A204028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20402C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A204030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A204034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 28. USART Channel 0 registers table

8.4.5.3.2 USART Channel 1 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A205000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A205004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A205008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.
TX_SADDR	0x1A205010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A205014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A205018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A205020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A205024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A205028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20502C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A205030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A205034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 29. USART Channel 1 registers table

8.4.5.3.3 USART Channel 2 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A206000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A206004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A206008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.
TX_SADDR	0x1A206010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A206014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A206018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A206020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A206024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A206028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20602C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A206030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A206034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 30. USART Channel 2 registers table

8.4.5.3.4 USART Channel 3 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A207000	32	Config	R/W	*0x0000*	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A207004	32	Config	R/W	*0x0000*	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A207008	32	Config	R/W	*0x0000*	uDMA RX UART stream configuration register.

Name	Address	Size	Type	Access	Default	Description
TX_SADDR	0x1A207010	32	Config	R/W	*0x0000*	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A207014	32	Config	R/W	*0x0000*	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A207018	32	Config	R/W	*0x0000*	uDMA TX UART stream configuration register.
STATUS	0x1A207020	32	Status	R	*0x0000*	uDMA UART status register.
SETUP	0x1A207024	32	Config	R/W	*0x0000*	UDMA UART configuration register.
ERROR	0x1A207028	32	Status	R	*0x0000*	uDMA UART Error status
IRQ_EN	0x1A20702C	32	Config	R/W	*0x0000*	uDMA UART Read or Error interrupt enable register.
VALID	0x1A207030	32	Status	R	*0x0000*	uDMA UART Read polling data valid flag register.
DATA	0x1A207034	32	Data	R	*0x0000*	uDMA UART Read polling data register.

Table 31. USART Channel 3 registers table

8.4.5.3.5 uDMA USART interface registers details

8.4.5.3.5.1 uDMA RX UART buffer base address configuration register. (RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 15:0 - **RX_SADDR** *(R/W)*

RX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

8.4.5.3.5.2 uDMA RX UART buffer size configuration register. (RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 16:0 - **RX_SIZE** *(R/W)*

RX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.3.5.3 uDMA RX UART stream configuration register. (RX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

RX channel clear and stop transfer:

- *0b0*: disable
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

RX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

RX channel enable and start transfer bitfield:

- *0b0*: disable
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

RX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.3.5.4 uDMA TX UART buffer base address configuration register. (TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 15:0 - **TX_SADDR** *(R/W)*

TX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

8.4.5.3.5.5 uDMA TX UART buffer size configuration register. (TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															TX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 16:0 - **TX_SIZE** *(R/W)*

TX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.3.5.6 uDMA TX UART stream configuration register. (TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

TX channel clear and stop transfer bitfield:

- *0b0*: disabled
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

TX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

TX channel enable and start transfer bitfield:

- *0b0*: disabled
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

TX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.3.5.7 uDMA UART status register. (STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														RX_BUSY	TX_BUSY

Bit 1 - **RX_BUSY** *(R)*

RX busy status flag:

- *0b0*: no RX transfer on-going
- *0b1*: RX transfer on-going

Bit 0 - **TX_BUSY** *(R)*

TX busy status flag:

- *0b0*: no TX transfer on-going
- *0b1*: TX transfer on-going

8.4.5.3.5.8 UDMA UART configuration register. (SETUP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKDIV															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RX_ENA	TX_ENA	RTS_EN	CTS_EN	CLEAN_FIFO	POLLING_EN	STOP_BITS	BIT_LENGTH		PARITY_ENA

Bits 31:16 - **CLKDIV** *(R/W)*

UART Clock divider configuration bitfield. The baudrate is equal to SOC_FREQ/CLKDIV.

Bit 9 - **RX_ENA** *(R/W)*

RX transceiver configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 8 - **TX_ENA** *(R/W)*

TX transceiver configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 7 - **RTS_EN** *(R/W)*

Request To Send (RTS) configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 6 - **CTS_EN** *(R/W)*

Clear To Send (CTS) configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

Bit 5 - **CLEAN_FIFO** *(R/W)*

In all mode clean the RX fifo, set 1 then set 0 to realize a reset fifo:

- *0b0*: Stop Clean the RX FIFO.
- *0b1*: Clean the RX FIFO.

Bit 4 - **POLLING_EN** *(R/W)*

When in uart read, use polling method to read the data, read interrupt enable flag will be ignored:

- *0b0*: Do not using polling method to read data.
- *0b1*: Using polling method to read data. Interrupt enable flag will be ignored.

Bit 3 - **STOP_BITS** *(R/W)*

Stop bits length bitfield:

- *0b0*: 1 stop bit
- *0b1*: 2 stop bits

Bits 2:1 - **BIT_LENGTH** *(R/W)*

Character length bitfield:

- *0b00*: 5 bits
- *0b01*: 6 bits
- *0b10*: 7 bits
- *0b11*: 8 bits

Bit 0 - **PARITY_ENA** *(R/W)*

Parity bit generation and check configuration bitfield:

- *0b0*: disabled
- *0b1*: enabled

8.4.5.3.5.9 uDMA UART Error status (ERROR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														RX_ERR_PARITY	RX_ERR_OVERFLOW

Bit 1 - **RX_ERR_PARITY** *(R)*

RX parity error status flag:

- *0b0*: no error
- *0b1*: RX parity error occurred

Bit 0 - **RX_ERR_OVERFLOW** *(R)*

RX overflow error status flag:

- *0b0*: no error
- *0b1*: RX overflow error occurred

8.4.5.3.5.10 uDMA UART Read or Error interrupt enable register. (IRQ_EN)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														ERROR	RX

Bit 1 - **ERROR** *(R/W)*

Error interrupt in enable flag:

- *0b0*: Error IRQ disable
- *0b1*: Error IRQ enable

Bit 0 - **RX** *(R/W)*

Rx interrupt in enable flag:

- *0b0*: RX IRQ disable
- *0b1*: RX IRQ enable

8.4.5.3.5.11 uDMA UART Read polling data valid flag register. (VALID)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															READY

Bit 0 - **READY** *(R)*

Used only in RX polling method to indicate data is ready for read:

- *0b0*: Data is not ready to read
- *0b1*: Data is ready to read

8.4.5.3.5.12 uDMA UART Read polling data register. (DATA)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BYTE							

Bits 7:0 - **BYTE** *(R)*

RX read data for polling or interrupt

8.4.5.4 uDMA SPIM interfaces

None

8.4.5.4.1 SPI Master Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A208000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A208004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A208008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A208010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A208014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A208018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A208020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A208024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A208028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 32. SPI Master Channel 0 registers table

8.4.5.4.2 SPI Master Channel 1 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A209000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A209004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A209008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A209010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A209014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A209018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A209020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A209024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer

Name	Address	Size	Type	Access	Default	Description
SPIM_CMD_CFG	0x1A209028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 33. SPI Master Channel 1 registers table

8.4.5.4.3 SPI Master Channel 2 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20A000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20A004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A20A008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20A010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20A014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20A018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20A020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20A024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20A028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 34. SPI Master Channel 2 registers table

8.4.5.4.4 SPI Master Channel 3 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20B000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20B004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A20B008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20B010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20B014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20B018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20B020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20B024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20B028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 35. SPI Master Channel 3 registers table

8.4.5.4.5 SPI Master Channel 4 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20C000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20C004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A20C008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20C010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20C014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20C018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20C020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20C024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20C028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 36. SPI Master Channel 4 registers table

8.4.5.4.6 SPI Master Channel 5 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20D000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20D004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_CFG	0x1A20D008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20D010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20D014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20D018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20D020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20D024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20D028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 37. SPI Master Channel 5 registers table

8.4.5.4.7 SPI Master Channel 6 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20E000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20E004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A20E008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20E010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20E014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20E018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20E020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20E024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20E028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 38. SPI Master Channel 6 registers table

8.4.5.4.8 SPI Master Channel 7 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A20F000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A20F004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A20F008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A20F010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A20F014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A20F018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A20F020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A20F024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A20F028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 39. SPI Master Channel 7 registers table

8.4.5.4.9 SPI Master Channel 8 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A210000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A210004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A210008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A210010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A210014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A210018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A210020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A210024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A210028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 40. SPI Master Channel 8 registers table

8.4.5.4.10 SPI Master Channel 9 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A211000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A211004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A211008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A211010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A211014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A211018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A211020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A211024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A211028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 41. SPI Master Channel 9 registers table

8.4.5.4.11 SPI Master Channel 10 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A212000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A212004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A212008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A212010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A212014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A212018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A212020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A212024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A212028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 42. SPI Master Channel 10 registers table

8.4.5.4.12 QSPI Master Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A213000	32	Config	R/W	*0x0000*	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A213004	32	Config	R/W	*0x0000*	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A213008	32	Config	R/W	*0x0004*	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A213010	32	Config	R/W	*0x0000*	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A213014	32	Config	R/W	*0x0000*	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A213018	32	Config	R/W	*0x0000*	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A213020	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A213024	32	Config	R/W	*0x0000*	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A213028	32	Config	R/W	*0x0004*	CMD SPI uDMA transfer configuration

Table 43. QSPI Master Channel 0 registers table

8.4.5.4.13 uDMA SPIM interface registers details

8.4.5.4.13.1 RX SPI uDMA transfer address of associated buffer (SPIM_RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX_SADDR															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 31:0 - **RX_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.4.13.2 RX SPI uDMA transfer size of buffer (SPIM_RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												RX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 19:0 - **RX_SIZE** *(R/W)*

Buffer size in bytes. (1MBytes maximum)

- Read: buffer size left
- Write: set buffer size

8.4.5.4.13.3 RX SPI uDMA transfer configuration (SPIM_RX_CFG)

Reset value: 0x0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved										CLR/PENDING	EN	Reserved	DATASIZE		CONTINOUS

Bit 5 - **CLR** *(W)*
Reset value: 0b0

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*
Reset value: 0b0

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*
Reset value: 0b0

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*

Reset value: 0b10

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)(default)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*

Reset value: 0b0

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.4.13.4 TX SPI uDMA transfer address of associated buffer (SPIM_TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TX_SADDR															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 31:0 - **TX_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.4.13.5 TX SPI uDMA transfer size of buffer (SPIM_TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 19:0 - **TX_SIZE** *(R/W)*

Buffer size in bytes. (1MBytes maximum)

- Read: buffer size left
- Write: set buffer size

8.4.5.4.13.6 TX SPI uDMA transfer configuration (SPIM_TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved										CLR/PENDING	EN	Reserved	DATASIZE		CONTINOUS

Bit 5 - **CLR** *(W)*

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)(default)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.4.13.7 CMD SPI uDMA transfer address of associated buffer (SPIM_CMD_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMD_SADDR															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD_SADDR															

Bits 31:0 - **CMD_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.4.13.8 CMD SPI uDMA transfer size of buffer (SPIM_CMD_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CMD_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD_SIZE															

Bits 19:0 - **CMD_SIZE** *(R/W)*

Buffer size in bytes. (1MBytes maximum)

- Read: buffer size left
- Write: set buffer size

8.4.5.4.13.9 CMD SPI uDMA transfer configuration (SPIM_CMD_CFG)

Reset value: 0x0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved										CLR/PENDING	EN	Reserved	DATASIZE		CONTINOUS

Bit 5 - **CLR** *(W)*

Reset value: 0b0

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*

Reset value: 0b0

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*

Reset value: 0b0

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*

Reset value: 0b10

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)(default)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*

Reset value: 0b0

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.5 uDMA I2C interfaces

None

8.4.5.5.1 I2C Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A214000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A214004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A214008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A214010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A214014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A214018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A214020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A214024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A214028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.
STATUS	0x1A214030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A214034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 44. I2C Channel 0 registers table

8.4.5.5.2 I2C Channel 1 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A215000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A215004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A215008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A215010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A215014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A215018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A215020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A215024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A215028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.
STATUS	0x1A215030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A215034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 45. I2C Channel 1 registers table

8.4.5.5.3 I2C Channel 2 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A216000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A216004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A216008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A216010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A216014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A216018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.

Name	Address	Size	Type	Access	Default	Description
CMD_SADDR	0x1A216020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A216024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A216028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.
STATUS	0x1A216030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A216034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 46. I2C Channel 2 registers table

8.4.5.5.4 I2C Channel 3 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A217000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A217004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A217008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A217010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A217014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A217018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A217020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A217024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A217028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.
STATUS	0x1A217030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A217034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 47. I2C Channel 3 registers table

8.4.5.5.5 I2C Channel 4 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A218000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A218004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A218008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A218010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A218014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A218018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A218020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A218024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A218028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.
STATUS	0x1A218030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A218034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 48. I2C Channel 4 registers table

8.4.5.5.6 I2C Channel 5 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A219000	32	Config	R/W	*0x0000*	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A219004	32	Config	R/W	*0x0000*	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A219008	32	Config	R/W	*0x0000*	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A219010	32	Config	R/W	*0x0000*	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A219014	32	Config	R/W	*0x0000*	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A219018	32	Config	R/W	*0x0000*	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A219020	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A219024	32	Config	R/W	*0x0000*	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A219028	32	Config	R/W	*0x0000*	uDMA CMD I2C stream configuration register.

Name	Address	Size	Type	Access	Default	Description
STATUS	0x1A219030	32	Status	R/W	*0x0000*	uDMA I2C Status register.
SETUP	0x1A219034	32	Config	R/W	*0x0000*	uDMA I2C Configuration register.

Table 49. I2C Channel 5 registers table

8.4.5.5.7 uDMA I2C interface registers details

8.4.5.5.7.1 uDMA RX I2C buffer base address configuration register. (RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											RX_SADDR				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 20:0 - **RX_SADDR** *(R/W)*

RX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

8.4.5.5.7.2 uDMA RX I2C buffer size configuration register. (RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												RX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 19:0 - **RX_SIZE** *(R/W)*

RX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.5.7.3 uDMA RX I2C stream configuration register. (RX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

RX channel clear and stop transfer:

- *0b0*: disable
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

RX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

RX channel enable and start transfer bitfield:

- *0b0*: disable
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

RX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.5.7.4 uDMA TX I2C buffer base address configuration register. (TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											TX_SADDR				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 20:0 - **TX_SADDR** *(R/W)*

TX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

8.4.5.5.7.5 uDMA TX I2C buffer size configuration register. (TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 19:0 - **TX_SIZE** *(R/W)*

TX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.5.7.6 uDMA TX I2C stream configuration register. (TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

TX channel clear and stop transfer bitfield:

- *0b0*: disabled
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

TX transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

TX channel enable and start transfer bitfield:

- *0b0*: disabled
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

TX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.5.7.7 uDMA CMD I2C buffer base address configuration register. (CMD_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											CMD_SADDR				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD_SADDR															

Bits 20:0 - **CMD_SADDR** *(R/W)*

CMD buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

8.4.5.5.7.8 uDMA CMD I2C buffer size configuration register. (CMD_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CMD_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD_SIZE															

Bits 19:0 - **CMD_SIZE** *(R/W)*

CMD buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.5.7.9 uDMA CMD I2C stream configuration register. (CMD_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved			CONTINOUS

Bit 6 - **CLR** *(W)*

CMD channel clear and stop transfer bitfield:

- *0b0*: disabled
- *0b1*: stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

CMD transfer pending in queue status flag:

- *0b0*: no pending transfer in the queue
- *0b1*: pending transfer in the queue

Bit 4 - **EN** *(R/W)*

CMD channel enable and start transfer bitfield:

- *0b0*: disabled
- *0b1*: enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

CMD channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.5.7.10 uDMA I2C Status register. (STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved													ACK	ARB_LOST	BUSY

Bit 2 - **ACK** *(R)*

I2C ack flag, can be polling for busy:

- *0b0*: ACK
- *0b1*: NAK

Bit 1 - **ARB_LOST** *(R/W)*

I2C arbitration lost status flag:

- *0b0*: no error
- *0b1*: arbitration lost error

Bit 0 - **BUSY** *(R/W)*

I2C bus busy status flag:

- *0b0*: no transfer on-going
- *0b1*: transfer on-going

8.4.5.5.7.11 uDMA I2C Configuration register. (SETUP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															DO_RST

Bit 0 - **DO_RST** *(R/W)*

Reset command used to abort the on-going transfer and clear busy and arbitration lost status flags.

8.4.5.6 uDMA SDIO interfaces

None

8.4.5.6.1 SDIO Channel 0 registers

Name	Address	Size	Type	Access	Default	Description
SDIO_RX_SADDR	0x1A21A000	32	Config	R/W	*0x0000*	RX SDIO uDMA transfer address of associated buffer
SDIO_RX_SIZE	0x1A21A004	32	Config	R/W	*0x0000*	RX SDIO uDMA transfer size of buffer
SDIO_RX_CFG	0x1A21A008	32	Config	R/W	*0x0004*	RX SDIO uDMA transfer configuration
SDIO_TX_SADDR	0x1A21A010	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer address of associated buffer
SDIO_TX_SIZE	0x1A21A014	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer size of buffer
SDIO_TX_CFG	0x1A21A018	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer configuration
SDIO_CMD_OP	0x1A21A020	32	Config	W	*0x0000*	SDIO command
SDIO_CMD_ARG	0x1A21A024	32	Config	W	*0x0000*	SDIO argument
SDIO_DATA_SETUP	0x1A21A028	32	Config	W	*0x0000*	Data transfer setup
SDIO_START	0x1A21A02C	32	Config	W	*0x0000*	Start
SDIO_RSP0	0x1A21A030	32	Config	R	*0x0000*	Response byte0
SDIO_RSP1	0x1A21A034	32	Config	R	*0x0000*	Response byte1
SDIO_RSP2	0x1A21A038	32	Config	R	*0x0000*	Response byte2
SDIO_RSP3	0x1A21A03C	32	Config	R	*0x0000*	Response byte3
SDIO_CLK_DIV	0x1A21A040	32	Config	R/W	*0x0000*	Clock Divider
SDIO_STATUS	0x1A21A044	32	Config	R/W	*0x0000*	STATUS
SDIO_STOPCMD_OP	0x1A21A048	32	Config	W	*0x0000*	SDIO STOP command op
SDIO_STOPCMD_ARG	0x1A21A052	32	Config	W	*0x0000*	SDIO STOP command arg

Table 50. SDIO Channel 0 registers table

8.4.5.6.2 SDIO Channel 1 registers

Name	Address	Size	Type	Access	Default	Description
SDIO_RX_SADDR	0x1A21B000	32	Config	R/W	*0x0000*	RX SDIO uDMA transfer address of associated buffer
SDIO_RX_SIZE	0x1A21B004	32	Config	R/W	*0x0000*	RX SDIO uDMA transfer size of buffer
SDIO_RX_CFG	0x1A21B008	32	Config	R/W	*0x0004*	RX SDIO uDMA transfer configuration
SDIO_TX_SADDR	0x1A21B010	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer address of associated buffer
SDIO_TX_SIZE	0x1A21B014	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer size of buffer
SDIO_TX_CFG	0x1A21B018	32	Config	R/W	*0x0000*	TX SDIO uDMA transfer configuration
SDIO_CMD_OP	0x1A21B020	32	Config	W	*0x0000*	SDIO command
SDIO_CMD_ARG	0x1A21B024	32	Config	W	*0x0000*	SDIO argument
SDIO_DATA_SETUP	0x1A21B028	32	Config	W	*0x0000*	Data transfer setup
SDIO_START	0x1A21B02C	32	Config	W	*0x0000*	Start
SDIO_RSP0	0x1A21B030	32	Config	R	*0x0000*	Response byte0
SDIO_RSP1	0x1A21B034	32	Config	R	*0x0000*	Response byte1
SDIO_RSP2	0x1A21B038	32	Config	R	*0x0000*	Response byte2
SDIO_RSP3	0x1A21B03C	32	Config	R	*0x0000*	Response byte3
SDIO_CLK_DIV	0x1A21B040	32	Config	R/W	*0x0000*	Clock Divider
SDIO_STATUS	0x1A21B044	32	Config	R/W	*0x0000*	STATUS
SDIO_STOPCMD_OP	0x1A21B048	32	Config	W	*0x0000*	SDIO STOP command op
SDIO_STOPCMD_ARG	0x1A21B052	32	Config	W	*0x0000*	SDIO STOP command arg

Table 51. SDIO Channel 1 registers table

8.4.5.6.3 uDMA SDIO interface registers details

8.4.5.6.3.1 RX SDIO uDMA transfer address of associated buffer (SDIO_RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX_SADDR															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 31:0 - **RX_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.6.3.2 RX SDIO uDMA transfer size of buffer (SDIO_RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												RX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 19:0 - **RX_SIZE** *(R/W)*

Buffer size in bytes. (1MBytes maximum)

- Read: buffer size left
- Write: set buffer size

8.4.5.6.3.3 RX SDIO uDMA transfer configuration (SDIO_RX_CFG)

Reset value: 0x0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved	DATASIZE		CONTINOUS

Bit 6 - **CLR** *(W)*
Reset value: 0b0

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*
Reset value: 0b0

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*
Reset value: 0b0

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*
Reset value: 0b10

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)(default)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*
Reset value: 0b0

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.6.3.4 TX SDIO uDMA transfer address of associated buffer (SDIO_TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TX_SADDR															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 31:0 - **TX_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.6.3.5 TX SDIO uDMA transfer size of buffer (SDIO_TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TX_SIZE			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 19:0 - **TX_SIZE** *(R/W)*

Buffer size in bytes. (1MBytes maximum)

- Read: buffer size left
- Write: set buffer size

8.4.5.6.3.6 TX SDIO uDMA transfer configuration (SDIO_TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved	DATASIZE		CONTINUOUS

Bit 6 - **CLR** *(W)*

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)(default)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.6.3.7 SDIO command (SDIO_CMD_OP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CMD_OP						Reserved					CMD_RSP_TYPE		

Bits 13:8 - **CMD_OP** *(W)*

SDIO command opcode

Bits 2:0 - **CMD_RSP_TYPE** *(W)*

Response type:

- 3'b000: No Responce
- 3'b001: 48 bits with CRC
- 3'b010: 48 bits no CRC
- 3'b011: 136bits
- 3'b100: 48 bits with BUSY check

8.4.5.6.3.8 SDIO argument (SDIO_CMD_ARG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMD_ARG															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD_ARG															

Bits 31:0 - **CMD_ARG** *(W)*

Argument to be sent with the command

8.4.5.6.3.9 Data transfer setup (SDIO_DATA_SETUP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						BLOCK_SIZE									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BLOCK_NUM								Reserved					DATA_QUAD	DATA_RWN	DATA_EN

Bits 25:16 - **BLOCK_SIZE** *(W)*

Sets the block size

Bits 15:8 - **BLOCK_NUM** *(W)*

Sets the number of blocks to be sent

Bit 2 - **DATA_QUAD** *(W)*

Enables QUAD mode

Bit 1 - **DATA_RWN** *(W)*

Selects the direction of transfer:

- *0b0*: write
- *0b1*: read

Bit 0 - **DATA_EN** *(W)*

Enables Data transfer for current command

8.4.5.6.3.10 Start (SDIO_START)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															START

Bit 0 - **START** *(W)*

Starts the SDIO transfer

8.4.5.6.3.11 Response byte0 (SDIO_RSP0)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDIO_RSP0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SDIO_RSP0															

Bits 31:0 - **SDIO_RSP0** *(R)*

Bytes0..3 of response

8.4.5.6.3.12 Response byte1 (SDIO_RSP1)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDIO_RSP1															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SDIO_RSP1															

Bits 31:0 - **SDIO_RSP1** *(R)*

Bytes4..7 of response

8.4.5.6.3.13 Response byte2 (SDIO_RSP2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDIO_RSP2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDIO_RSP2															

Bits 31:0 - **SDIO_RSP2** *(R)*

Bytes8..11 of response

8.4.5.6.3.14 Response byte3 (SDIO_RSP3)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDIO_RSP3															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDIO_RSP3															

Bits 31:0 - **SDIO_RSP3** *(R)*

Bytes12..15 of response

8.4.5.6.3.15 Clock Divider (SDIO_CLK_DIV)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CLK_DIV								

Bits 8:0 - **CLK_DIV** *(W)*

Clock Divider

8.4.5.6.3.16 STATUS (SDIO_STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DATA_ERR_STATUS						Reserved		CMD_ERR_STATUS					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ERROR	EOT

Bits 29:24 - **DATA_ERR_STATUS** *(R)*

Indicate the error status of the transfer of data :

- 6'b000000: no error
- *0b000001*: Response Time Out

Bits 21:16 - **CMD_ERR_STATUS** *(R)*

Indicate the error status of the transfer of command :

- 6'b000000: no error
- *0b000001*: Response Time Out
- *0b000010*: Response Wrong Direction
- *0b000100*: Response Busy Timeout

Bit 1 - **ERROR** *(R/W)*

Indicate the error of the transfer of command or data :

- *0b0*: no error
- *0b1*: error

Bit 0 - **EOT** *(R/W)*

Indicate the end of the transfer of command or data :

- *0b0*: not end
- *0b1*: end

8.4.5.6.3.17 SDIO STOP command op (SDIO_STOPCMD_OP)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		STOPCMD_OP						Reserved					STOPCMD_RSP_TYPE		

Bits 13:8 - **STOPCMD_OP** *(W)*

SDIO STOP command opcode

Bits 2:0 - **STOPCMD_RSP_TYPE** *(W)*

SDIO STOP command response type:

- 3'b000: No Responce
- 3'b001: 48 bits with CRC
- 3'b010: 48 bits no CRC
- 3'b011: 136bits
- 3'b100: 48 bits with BUSY check

8.4.5.6.3.18 SDIO STOP command arg (SDIO_STOPCMD_ARG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STOPCMD_ARG															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STOPCMD_ARG															

Bits 31:0 - **STOPCMD_ARG** *(W)*

Argument to be sent with the STOP command

8.4.5.6.4 uDMA SDIO interface commands

Name	Command number	Size	Description
CMD_GO_IDLE_STATE	0	6	powerup or reset card to idle state
CMD_SEND_CID	2	6	instruct all cards to send CID
CMD_SEND_RCA	3	6	instruct all cards to send RCA
ACMD_CFG_QUAD	6	6	configure the number of lines to use (single or quad mode)
CMD_SELECT	7	6	Select one card with its RCA and flip READY state
CMD_SEND_VOLTAGE	8	6	Send host voltage configuration
CMD_SD_STOP	18	6	STOP command for sd cards, stop multi block transfers after current block
CMD_READ_SINGLE_BLOCK	23	6	Ask the device for one single block of data
CMD_READ_MULT_BLOCK	24	6	Ask the device to continuously send blocks of data until it receives stop
CMD_WRITE_SINGLE_BLOCK	36	6	Write a single block of data to the device
CMD_WRITE_MULT_BLOCK	37	6	Write a continuous stream of data blocks to device
ACMD_HCS	65	6	Send host capacity support (high or standard capacity) and ask device for its voltage window
CMD_APP_SPEC_CMD	85	6	Signal next command is a specific command

Table 52. uDMA SDIO interface commands table

8.4.5.6.5 uDMA SDIO interface commands details

8.4.5.6.5.1 powerup or reset card to idle state (CMD_GO_IDLE_STATE)

Command number: 0

This command is followed by extra parameter bytes that are long. none

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.2 instruct all cards to send CID (CMD_SEND_CID)

Command number: 2

This command is followed by extra parameter bytes that are long. none

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.3 instruct all cards to send RCA (CMD_SEND_RCA)

Command number: 3

This command is followed by extra parameter bytes that are long. none

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.4 configure the number of lines to use (single or quad mode) (ACMD_CFG_QUAD)

Command number: 6

This command is followed by extra parameter bytes that are long. nb data lanes

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.5 Select one card with its RCA and flip READY state (CMD_SELECT)

Command number: 7

This command is followed by extra parameter bytes that are long. RCA

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.6 Send host voltage configuration (CMD_SEND_VOLTAGE)

Command number: 8

This command is followed by extra parameter bytes that are long. host voltage range

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.7 STOP command for sd cards, stop multi block transfers after current block (CMD_SD_STOP)

Command number: 18

This command is followed by extra parameter bytes that are long. none

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.8 Ask the device for one single block of data (CMD_READ_SINGLE_BLOCK)

Command number: 23

This command is followed by extra parameter bytes that are long. device address

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.9 Ask the device to continuously send blocks of data until it receives stop (CMD_READ_MULT_BLOCK)

Command number: 24

This command is followed by extra parameter bytes that are long. device address

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.10 Write a single block of data to the device (CMD_WRITE_SINGLE_BLOCK)

Command number: 36

This command is followed by extra parameter bytes that are long. device address

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.11 Write a continuous stream of data blocks to device (CMD_WRITE_MULT_BLOCK)

Command number: 37

This command is followed by extra parameter bytes that are long. device address

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.12 Send host capacity support (high or standard capacity) and ask device for its voltage window (ACMD_HCS)

Command number: 65

This command is followed by extra parameter bytes that are long. capacity mode

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.6.5.13 Signal next command is a specific command (CMD_APP_SPEC_CMD)

Command number: 85

This command is followed by extra parameter bytes that are long. RCA

5	4	3	2	1	0
CMD_OP	Reserved		CMD_RSP		

Bits 13:8 - **CMD_OP**

Bits 2:0 - **CMD_RSP**

8.4.5.7 uDMA CAM CPI interfaces

None

8.4.5.7.1 CAM channel 0 registers

Name	Address	Size	Type	Access	Default	Description
CAM_RX_SADDR	0x1A21C000	32	Config	R/W	*0x0000*	RX Camera uDMA transfer address of associated buffer register
CAM_RX_SIZE	0x1A21C004	32	Config	R/W	*0x0000*	RX Camera uDMA transfer size of buffer register
CAM_RX_CFG	0x1A21C008	32	Config	R/W	*0x0000*	RX Camera uDMA transfer configuration register
CAM_CFG_GLOB	0x1A21C020	32	Config	R/W	*0x0000*	Global configuration register
CAM_CFG_LL	0x1A21C024	32	Config	R/W	*0x0000*	Lower Left corner configuration register
CAM_CFG_UR	0x1A21C028	32	Config	R/W	*0x0000*	Upper Right corner configuration register
CAM_CFG_SIZE	0x1A21C02C	32	Config	R/W	*0x0000*	Horizontal Resolution configuration register
CAM_CFG_FILTER	0x1A21C030	32	Config	R/W	*0x0000*	RGB coefficients configuration register
CAM_VSYNC_POLARITY	0x1A21C034	32	Config	R/W	*0x0000*	VSYNC Polarity register

Table 53. CAM channel 0 registers table

8.4.5.7.2 CAM channel 1 registers

Name	Address	Size	Type	Access	Default	Description
CAM_RX_SADDR	0x1A21D000	32	Config	R/W	*0x0000*	RX Camera uDMA transfer address of associated buffer register
CAM_RX_SIZE	0x1A21D004	32	Config	R/W	*0x0000*	RX Camera uDMA transfer size of buffer register
CAM_RX_CFG	0x1A21D008	32	Config	R/W	*0x0000*	RX Camera uDMA transfer configuration register
CAM_CFG_GLOB	0x1A21D020	32	Config	R/W	*0x0000*	Global configuration register
CAM_CFG_LL	0x1A21D024	32	Config	R/W	*0x0000*	Lower Left corner configuration register
CAM_CFG_UR	0x1A21D028	32	Config	R/W	*0x0000*	Upper Right corner configuration register
CAM_CFG_SIZE	0x1A21D02C	32	Config	R/W	*0x0000*	Horizontal Resolution configuration register
CAM_CFG_FILTER	0x1A21D030	32	Config	R/W	*0x0000*	RGB coefficients configuration register
CAM_VSYNC_POLARITY	0x1A21D034	32	Config	R/W	*0x0000*	VSYNC Polarity register

Table 54. CAM channel 1 registers table

8.4.5.7.3 uDMA CAM CPI interface registers details

8.4.5.7.3.1 RX Camera uDMA transfer address of associated buffer register (CAM_RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 15:0 - **RX_SADDR** *(R/W)*

Configure pointer to memory buffer:

- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

8.4.5.7.3.2 RX Camera uDMA transfer size of buffer register (CAM_RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 16:0 - **RX_SIZE** *(R/W)*

Buffer size in bytes. (128kBytes maximum)

- Read: buffer size left
- Write: set buffer size

NOTE: Careful with size in byte. If you use uncompressed pixel data mapped on 16 bits, you have to declare buffer size in bytes even if buffer type is short.

8.4.5.7.3.3 RX Camera uDMA transfer configuration register (CAM_RX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved									CLR	PENDING	EN	Reserved	DATASIZE		CONTINOUS

Bit 6 - **CLR** *(W)*

Channel clear and stop transfer:

- *0b0*: disable
- *0b1*: enable

Bit 5 - **PENDING** *(R)*

Transfer pending in queue status flag:

- *0b0*: free
- *0b1*: pending

Bit 4 - **EN** *(R/W)*

Channel enable and start transfer:

- *0b0*: disable
- *0b1*: enable

This signal is used also to queue a transfer if one is already ongoing.

Bits 2:1 - **DATASIZE** *(R/W)*

Channel transfer size used to increment uDMA buffer address pointer:

- *0b00*: +1 (8 bits)
- *0b01*: +2 (16 bits)
- *0b10*: +4 (32 bits)
- *0b11*: +0

Bit 0 - **CONTINOUS** *(R/W)*

Channel continuous mode:

- *0b0*: disable
- *0b1*: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

8.4.5.7.3.4 Global configuration register (CAM_CFG_GLOB)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	Reserved														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	SHIFT				FORMAT			FRAME SLICE_EN	FRAMEDROP_VAL						FRAME DROP_EN

Bit 31 - **EN** *(R/W)*

Enable data rx from camera interface.

The enable/disable happens only at the start of a frame.

- *0b0*: disable
- *0b1*: enable

Bits 14:11 - **SHIFT** *(R/W)*

Right shift of final pixel value (DivFactor)

NOTE: not used if FORMAT == BYPASS

Bits 10:8 - **FORMAT** *(R/W)*

Input frame format:

- *0b000*: RGB565
- *0b001*: RGB555
- *0b010*: RGB444
- *0b100*: BYPASS_LITEND
- 3'b101: BYPASS_BIGEND

Bit 7 - **FRAMESLICE_EN** *(R/W)*

Input frame slicing:

- *0b0*: disable
- *0b1*: enable

Bits 6:1 - **FRAMEDROP_VAL** *(R/W)*

Sets how many frames should be dropped after each received.

Bit 0 - **FRAMEDROP_EN** *(R/W)*

Frame dropping:

- *0b0*: disable
- *0b1*: enable

8.4.5.7.3.5 Lower Left corner configuration register (CAM_CFG_LL)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRAMESLICE_LLY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAMESLICE_LLX															

Bits 31:16 - **FRAMESLICE_LLY** *(R/W)*

Y coordinate of lower left corner of slice

Bits 15:0 - **FRAMESLICE_LLX** *(R/W)*

X coordinate of lower left corner of slice

8.4.5.7.3.6 Upper Right corner configuration register (CAM_CFG_UR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRAMESLICE_URY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAMESLICE_URX															

Bits 31:16 - **FRAMESLICE_URY** *(R/W)*

Y coordinate of upper right corner of slice

Bits 15:0 - **FRAMESLICE_URX** *(R/W)*

X coordinate of upper right corner of slice

8.4.5.7.3.7 Horizontal Resolution configuration register (CAM_CFG_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ROWLEN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits 31:16 - **ROWLEN** *(R/W)*

Horizontal Resolution. It is used for slice mode. Value set into the bitfield must be equal to (rowlen-1).

8.4.5.7.3.8 RGB coefficients configuration register (CAM_CFG_FILTER)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								R_COEFF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G_COEFF								B_COEFF							

Bits 23:16 - **R_COEFF** *(R/W)*

Coefficient that multiplies the R component

NOTE: not used if FORMAT == BYPASS

Bits 15:8 - **G_COEFF** *(R/W)*

Coefficient that multiplies the G component

NOTE: not used if FORMAT == BYPASS

Bits 7:0 - **B_COEFF** *(R/W)*

Coefficient that multiplies the B component

NOTE: not used if FORMAT == BYPASS

8.4.5.7.3.9 VSYNC Polarity register (CAM_VSYNC_POLARITY)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															VSYNC_ POLARI TY

Bit 0 - **VSYNC_POLARITY** *(R/W)*

Set vsync polarity of CPI.

- *0b0*: Active 0
- *0b1*: Active 1

8.4.5.8 uDMA Filter Interface

None

8.4.5.8.1 Filter registers

Name	Address	Size	Type	Access	Default	Description
REG_TX_CH0_ADD	0x1A21E000	32	Config	R/W	*0x0000*	FILTER tx channel 0 address register
REG_TX_CH0_CFG	0x1A21E004	32	Config	R/W	*0x0000*	FILTER tx channel 0 configuration register
REG_TX_CH0_LEN0	0x1A21E008	32	Config	R/W	*0x0000*	FILTER tx channel 0 length0 register
REG_TX_CH0_LEN1	0x1A21E00C	32	Config	R/W	*0x0000*	FILTER tx channel 0 length1 register
REG_TX_CH0_LEN2	0x1A21E010	32	Config	R/W	*0x0000*	FILTER tx channel 0 length2 register
REG_TX_CH1_ADD	0x1A21E014	32	Config	R/W	*0x0000*	FILTER tx channel 1 address register
REG_TX_CH1_CFG	0x1A21E018	32	Config	R/W	*0x0000*	FILTER tx channel 1 configuration register
REG_TX_CH1_LEN0	0x1A21E01C	32	Config	R/W	*0x0000*	FILTER tx channel 1 length0 register
REG_TX_CH1_LEN1	0x1A21E020	32	Config	R/W	*0x0000*	FILTER tx channel 1 length1 register
REG_TX_CH1_LEN2	0x1A21E024	32	Config	R/W	*0x0000*	FILTER tx channel 1 length2 register
REG_RX_CH_ADD	0x1A21E028	32	Config	R/W	*0x0000*	FILTER RX channel address register

Name	Address	Size	Type	Access	Default	Description
REG_RX_CH_CFG	0x1A21E02C	32	Config	R/W	*0x0000*	FILTER RX channel configuration register
REG_RX_CH_LEN0	0x1A21E030	32	Config	R/W	*0x0000*	FILTER RX channel length0 register
REG_RX_CH_LEN1	0x1A21E034	32	Config	R/W	*0x0000*	FILTER RX channel length1 register
REG_RX_CH_LEN2	0x1A21E038	32	Config	R/W	*0x0000*	FILTER RX channel length2 register
REG_AU_CFG	0x1A21E03C	32	Config	R/W	*0x0000*	FILTER arithmetic unit configuration register
REG_AU_REG0	0x1A21E040	32	Config	R/W	*0x0000*	FILTER arithmetic unit 0 register
REG_AU_REG1	0x1A21E044	32	Config	R/W	*0x0000*	FILTER arithmetic unit 1 register
REG_BINCU_TH	0x1A21E048	32	Config	R/W	*0x0000*	FILTER binarization threshold register
REG_BINCU_CNT	0x1A21E04C	32	Config	R/W	*0x0000*	FILTER binarization count register
REG_BINCU_SETUP	0x1A21E050	32	Config	R/W	*0x0000*	FILTER binarization datasize format register
REG_BINCU_VAL	0x1A21E054	32	Status	R	*0x0000*	FILTER binarization result count register
REG_FILT	0x1A21E058	32	Config	R/W	*0x0000*	FILTER control mode register
REG_FILT_CMD	0x1A21E05C	32	Config	R/W	*0x0000*	FILTER start register
REG_STATUS	0x1A21E060	32	Status	R/W	*0x0000*	FILTER status register

Table 55. Filter registers table

8.4.5.8.2 uDMA Filter Interface registers details

8.4.5.8.2.1 FILTER tx channel 0 configuration register (REG_TX_CH0_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						MODE		Reserved						SIZE	

Bits 9:8 - **MODE** *(R)*

Data transfer mode:

- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

Bits 1:0 - **SIZE** *(R)*

Data transfer format:

- 2'b00: 8-bit
- 2'b01:16-bit
- 2;b10:32-bit

8.4.5.8.2.2 FILTER tx channel 1 configuration register (REG_TX_CH1_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						MODE		Reserved						SIZE	

Bits 9:8 - **MODE** *(R/W)*

Data transfer mode:

- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

Bits 1:0 - **SIZE** *(R)*

Data transfer format:

- 2'b00: 8-bit
- 2'b01:16-bit
- 2'b10:32-bit

8.4.5.8.2.3 FILTER RX channel configuration register (REG_RX_CH_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						MODE		Reserved						SIZE	

Bits 9:8 - **MODE** *(R/W)*

Data transfer mode:

- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

Bits 1:0 - **SIZE** *(R/W)*

Data transfer format:

- 2'b00: 8-bit
- 2'b01:16-bit
- 2;b10:32-bit

8.4.5.8.2.4 FILTER arithmetic unit configuration register (REG_AU_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											SHIFT				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				MODE				Reserved						BYPASS	SIGNED

Bits 20:16 - **SHIFT** *(R/W)*

Arithmetic Unit shift window size, (0 – 31).

Bits 11:8 - **MODE** *(R/W)*

Arithmetic Unit mode:

- 4'b0000: AU_MODE_AxB
- 4'b0001: AU_MODE_AxB+REG0
- 4'b0010: AU_MODE_AxB accumulation
- 4'b0011: AU_MODE_AxA
- 4'b0100: AU_MODE_AxA+B
- 4'b0101: AU_MODE_AxA-B
- 4'b0110: AU_MODE_AxA accumulation
- 4'b0111: AU_MODE_AxA+REG0
- 4'b1000: AU_MODE_AxREG1
- 4'b1001: AU_MODE_AxREG1+B
- 4'b1010: AU_MODE_AxREG1-B
- 4'b1011: AU_MODE_AxREG1+REG0
- 4'b1100: AU_MODE_AxREG1 accumulation
- 4'b1101: AU_MODE_A+B
- 4'b1110: AU_MODE_A-B
- 4'b1111: AU_MODE_A+REG0

Bit 1 - **BYPASS** *(R/W)*

Arithmetic Unit bypass or not.

- 1'b0: not bypass AU
- 1'b1: bypass AU

Bit 0 - **SIGNED** *(R/W)*

Arithmetic Unit result signed or not.

- 1'b0: not signed
- 1'b1: signed

8.4.5.8.2.5 FILTER binarization count register (REG_BINCU_CNT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	Reserved											COUNT			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COUNT															

Bit 31 - **EN** *(R/W)*

Binarization and counting unit enable:

- 1'b0: not enable
- 1'b1: enable

Bits 19:0 - **COUNT** *(R/W)*

Binarization and counting unit count value set.

8.4.5.8.2.6 FILTER start register (REG_FILT_CMD)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															START

Bit 0 - **START** *(R/W)*

Filter start flag, write only, write 1 to start the filter :

8.4.5.8.2.7 FILTER status register (REG_STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DONE

Bit 0 - **DONE** *(R/W)*

Filter done flag, write 1 to clear the flag :

- 1'b0: Filter process is not finished
- 1'b1: Filter process is finished

8.4.5.9 uDMA HYPERBUS Registerfile 0

None

8.4.5.9.1 HYPERBUS Reister file 0 registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A21F000	32	Config	R/W	*0x0000*	uDMA RX HYPERBUS buffer base address configuration register.
RX_SIZE	0x1A21F004	32	Config	R/W	*0x0000*	uDMA RX HYPERBUS buffer size configuration register.
RX_CFG	0x1A21F008	32	Config	R/W	*0x0000*	uDMA RX HYPERBUS stream configuration register.
TX_SADDR	0x1A21F00C	32	Config	R/W	*0x0000*	uDMA TX HYPERBUS buffer base address configuration register.
TX_SIZE	0x1A21F010	32	Config	R/W	*0x0000*	uDMA TX HYPERBUS buffer size configuration register.
TX_CFG	0x1A21F014	32	Config	R/W	*0x0000*	uDMA TX HYPERBUS stream configuration register.
CA_SETUP	0x1A21F018	32	Config	R/W	*0x0005*	Command-Addres Setup
HYPER_ADDR	0x1A21F01C	32	Config	R/W	*0x0000*	Set address in a hyper ram
HYPER_CFG	0x1A21F020	32	Config	R/W	*0x0000*	Set the configuration data for HyperRAM
STATUS	0x1A21F024	32	Status	R	*0x0000*	Status Register
TWD_ACT_EXT	0x1A21F028	32	Config	R/W	*0x0000*	Set 2D transfer activation
TWD_COUNT_EXT	0x1A21F02C	32	Config	R/W	*0x0000*	Set 2D transfer count
TWD_STRIDE_EXT	0x1A21F030	32	Config	R/W	*0x0000*	Set 2D transfer stride
TWD_ACT_L2	0x1A21F034	32	Config	R/W	*0x0000*	Set 2D transfer activation
TWD_COUNT_L2	0x1A21F038	32	Config	R/W	*0x0000*	set 2D transfer count
TWD_STRIDE_L2	0x1A21F03C	32	Config	R/W	*0x0000*	Set 2D transfer stride

Table 56. HYPERBUS Reister file 0 registers table

8.4.5.9.2 uDMA HYPERBUS Registerfile 0 registers details

8.4.5.9.2.1 uDMA RX HYPERBUS buffer base address configuration register. (RX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SADDR															

Bits 15:0 - **RX_SADDR** *(R/W)*

RX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

8.4.5.9.2.2 uDMA RX HYPERBUS buffer size configuration register. (RX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															RX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX_SIZE															

Bits 16:0 - **RX_SIZE** *(R/W)*

RX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.9.2.3 uDMA RX HYPERBUS stream configuration register. (RX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved										CLR/PENDING	EN	Reserved			CONTINOUS

Bit 5 - **CLR** *(W)*

RX channel clear and stop transfer:

- *0b0*: disable
- *0b1*: stop and clear - stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

RX transfer pending in queue status flag:

- *0b0*: no pending - no pending transfer in the queue
- *0b1*: pending - pending transfer in the queue

Bit 4 - **EN** *(R/W)*

RX channel enable and start transfer bitfield:

- *0b0*: disable
- *0b1*: start - enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

RX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.9.2.4 uDMA TX HYPERBUS buffer base address configuration register. (TX_SADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SADDR															

Bits 15:0 - **TX_SADDR** *(R/W)*

TX buffer base address bitfield:

- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

8.4.5.9.2.5 uDMA TX HYPERBUS buffer size configuration register. (TX_SIZE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															TX_SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_SIZE															

Bits 16:0 - **TX_SIZE** *(R/W)*

TX buffer size bitfield in bytes. (128kBytes maximum)

- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

8.4.5.9.2.6 uDMA TX HYPERBUS stream configuration register. (TX_CFG)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved										CLR/PENDING	EN	Reserved			CONTINOUS

Bit 5 - **CLR** *(W)*

TX channel clear and stop transfer bitfield:

- *0b0*: disabled
- *0b1*: stop and clear - stop and clear the on-going transfer

Bit 5 - **PENDING** *(R)*

TX transfer pending in queue status flag:

- *0b0*: no pending - no pending transfer in the queue
- *0b1*: pending - pending transfer in the queue

Bit 4 - **EN** *(R/W)*

TX channel enable and start transfer bitfield:

- *0b0*: disabled
- *0b1*: start - enable and start the transfer

This signal is used also to queue a transfer if one is already ongoing.

Bit 0 - **CONTINOUS** *(R/W)*

TX channel continuous mode bitfield:

- *0b0*: disabled
- *0b1*: enabled

At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

8.4.5.9.2.7 Command-Addres Setup (CA_SETUP)

Reset value: 0x0005

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved													RW	ADDR_SPACE	BURST_TYPE

Bit 2 - **RW** *(R/W)*

Reset value: 0b1

Read Write

0: write operation

1: read operation

Bit 1 - **ADDR_SPACE** *(R/W)*

Reset value: 0b0

Addr_space

0: memory array

1: register space

Bit 0 - **BURST_TYPE** *(R/W)*

Reset value: 0b1

Burst_type

0: wrapped burst (Not supported)

1: Linear burst

8.4.5.9.2.8 Set address in a hyper ram (HYPER_ADDR)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HYPER_ADDR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HYPER_ADDR															

Bits 31:0 - **HYPER_ADDR** *(R/W)*

Address of External memory

8.4.5.9.2.9 Status Register (STATUS)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											TRANS_WAITING				BUSY

Bits 4:1 - **TRANS_WAITING** *(R)*

ID transaction waiting

Bit 0 - **BUSY** *(R)*

Busy:

0 : Not Busy

1: Busy

8.4.5.9.2.10 Set 2D transfer activation (TWD_ACT_EXT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TWD_ACT_EXT

Bit 0 - **TWD_ACT_EXT** *(R/W)*

Enable

0: 2D transaction is disabled

1: 2D transaction is activated

8.4.5.9.2.11 Set 2D transfer count (TWD_COUNT_EXT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TWD_COUNT_EXT			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWD_COUNT_EXT															

Bits 19:0 - **TWD_COUNT_EXT** *(R/W)*

2D count length (in byte)

8.4.5.9.2.12 Set 2D transfer stride (TWD_STRIDE_EXT)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TWD_STRIDE_EXT			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWD_STRIDE_EXT															

Bits 19:0 - **TWD_STRIDE_EXT** *(R/W)*

2D stride length (in byte)

8.4.5.9.2.13 Set 2D transfer activation (TWD_ACT_L2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TWD_ACT_L2

Bit 0 - **TWD_ACT_L2** *(R/W)*

8.4.5.9.2.14 set 2D transfer count (TWD_COUNT_L2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TWD_COUNT_L2			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWD_COUNT_L2															

Bits 19:0 - **TWD_COUNT_L2** *(R/W)*

8.4.5.9.2.15 Set 2D transfer stride (TWD_STRIDE_L2)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TWD_STRIDE_L2			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWD_STRIDE_L2															

Bits 19:0 - **TWD_STRIDE_L2** *(R/W)*

8.4.5.10 uDMA HYPERBUS Registerfile 1

None

8.4.5.10.1 HYPERBUS Reister file 1 registers

Name	Address	Size	Type	Access	Default	Description
PAGE_BOUND	0x1A220000	32	Config	R/W	*0x0000*	Page boundary setting for the external memory
T_LATENCY_ACCESS	0x1A220004	32	Config	R/W	*0x0000*	The latency count of the Hyper Bus protocol
EN_LATENCY_ADD	0x1A220008	32	Config	R/W	*0x0000*	Enable Additional latency
T_CS_MAX	0x1A22000C	32	Config	R/W	*0x0000*	Maximum cycle counts for negating the chip select signal
T_RW_RECOVERY	0x1A220010	32	Config	R/W	*0x0000*	Cycle counts for T read write recovery of the Hyper bus protocol
T_RWDS_DELAY_LINE	0x1A220014	32	Config	R/W	*0x0000*	RWDS Delay Line
T_VARI_LATENCY	0x1A220018	32	Config	R/W	*0x0005*	Cycle counts for capturing the input rwds signa
N_HYPER_DEVICE	0x1A22001C	32	Config	R/W	*0x0000*	Set the number of connected devices
MEM_SEL	0x1A220020	32	Config	R/W	*0x0000*	Set Memory type
TRANS_ID_ALLOC	0x1A220024	32	Status	R	*0x0000*	Set 2D transfer stride
CHIPSEL_SEL	0x1A220028	32	Config	R/W	*0x0000*	Set which chip select to lower in the transaction

Table 57. HYPERBUS Reister file 1 registers table

8.4.5.10.2 uDMA HYPERBUS Registerfile 1 registers details

8.4.5.10.2.1 Page boundary setting for the external memory (PAGE_BOUND)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PAGE_BOUND		

Bits 2:0 - **PAGE_BOUND** *(R/W)*

Page boundary setting for the external memory

3'b000: 128 bytes

3'b001: 256 bytes

3'b010: 512 bytes

3'b011: 1024 bytes

Others: no_boundary

8.4.5.10.2.2 The latency count of the Hyper Bus protocol (T_LATENCY_ACCESS)

Reset value: 0x0000

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											T_LATENCY_ACCESS				

Bits 4:0 - **T_LATENCY_ACCESS** *(R/W)*

The latency count of the Hyper Bus protocol

8.4.5.10.2.3 Enable Additional latency (EN_LATENCY_ADD)

Reset value: 0x0000

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN_LATENCY_ADD

Bit 0 - **EN_LATENCY_ADD** *(R/W)*

It depends on Hyper bus rwds signal whether or not the additional latency of the hyper bus protocol is added.

1: force the module to have the additional latency

8.4.5.10.2.4 Maximum cycle counts for negating the chip select signal (T_CS_MAX)

Reset value: 0x0000

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T_CS_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T_CS_MAX															

Bits 31:0 - **T_CS_MAX** *(R/W)*

Maximum cycle counts for negating the chip select signal

8.4.5.10.2.5 Cycle counts for T read write recovery of the Hyper bus protocol (T_RW_RECOVERY)

Reset value: 0x0000

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
T_RW_RECOVERY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T_RW_RECOVERY															

Bits 31:0 - **T_RW_RECOVERY** *(R/W)*

Cycle counts for T read write recovery of the Hyper bus protocol

8.4.5.10.2.6 RWDS Delay Line (T_RWDS_DELAY_LINE)

Reset value: 0x0000

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												T_RWDS_DELAY_LINE			

Bits 3:0 - **T_RWDS_DELAY_LINE** *(R/W)*

Configuration for the RWDS delay line

8.4.5.10.2.7 Cycle counts for capturing the input rwds signa (T_VARI_LATENCY)

Reset value: 0x0005

Not connected to the PHY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												T_VARI_LATENCY			

Bits 3:0 - **T_VARI_LATENCY** *(R/W)*

Reset value: 0b0101

Cycle counts for capturing the input rwds signal

8.4.5.10.2.8 Set the number of connected devices (N_HYPER_DEVICE)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													N_HYPER_DEVICE		

Bits 2:0 - **N_HYPER_DEVICE** *(R/W)*

8.4.5.10.2.9 Set Memory type (MEM_SEL)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MEM_SEL	

Bits 1:0 - **MEM_SEL** *(R/W)*

Set Memory select:

2’b00: Hyper RAM

2’b01: Hyper Flash

2’b10: PSRAM

8.4.5.10.2.10 Set 2D transfer stride (TRANS_ID_ALLOC)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TRANS_ID_ALLOC

Bit 0 - **TRANS_ID_ALLOC** *(R)*

8.4.5.10.2.11 Set which chip select to lower in the transaction (CHIPSEL_SEL)

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											CHIPSEL_SEL				

Bits 4:0 - **CHIPSEL_SEL** *(R/W)*