GAP9 Datasheet

Version 0.10
Confidential information – Shared under NDA

Important note: GAP9 is subject to EU export restrictions according to Council Regulation (EC) No. 428/2009, dual-use control category 5A002.a. Non-reversible hardware disablement of Quiddikey and AES cryptography features through eFuse configuration allows to lift this restriction if needed. Contact GreenWaves Technologies for more information.

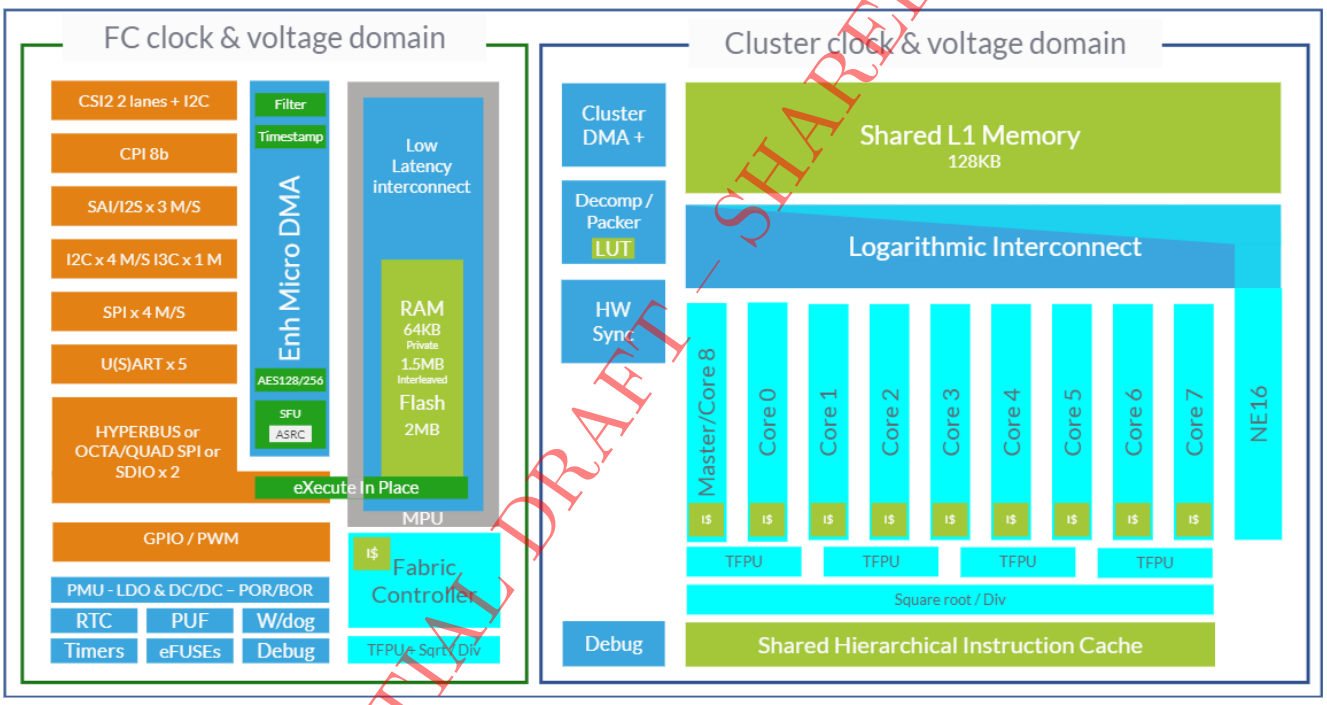
Change log for this release is available at the end of this document, in the Revisions chapter.

Table of contents

Chapter 1

Features

1.1 Device Overview



1.2 Performance

- System performance of 330µW/GOP
- Up to 370MHz internal clock
- Cluster executes 72 GOPS at a few tens of mWs
- FC executes 2 GOPS at a few mWs
- NE16 executes 150 8-bit MACs/cycle including load & store – 150GOPS, 25.6GB/sec data interface
- 25µA deep sleep current
- Selective retention on all L2 memory
- 0.8V down to 0.65V internal core VDD supply
- Integrated LDO / DC-DC
- 1.8V – I/O voltage
- 1.8V – 5.5V regulator supply voltage
- 0.4ms (TBC) cold boot time

- 2µs (TBC) to power and start cluster
- 1.5mW low power awake state with wake-up time of a few microseconds

1.3 Architecture Efficiency

- 10 identical high performance, extended ISA, RISC-V ISA cores
- Dynamic voltage & frequency scaling and automatic body biasing
- Multiple power states: deep sleep, deep sleep with retentive RAM, low activity, SOC on, SOC on & cluster on
- Fabric controller (FC) core for control and communication
- Cluster of 9 cores for compute-intensive tasks:
  - Logarithmic interconnect
  - Hardware event synchronization
  - Hierarchical, shared instruction cache
  - NE16 neural networks accelerator
- Software controlled explicit data movement across the memory hierarchy
  - Multi-channel 1D/2D cluster DMA with integrated data decompressor / packer / unpacker
  - Specialized multi-channel micro-DMA for autonomous peripheral support (hardware encryption, timestamping…)
  - Smart Filtering Unit – Programmable, low latency (1µs SAI to SAI delay for Active Noise Cancellation) filtering processor

1.4 Hardware Features

- Fabric controller core: 2kB instruction cache
- Cluster:
  - 128kB shared data memory
  - Hierarchical instruction cache: 4kB shared, 6kB private (512B per core except core 8: 2kB)
- All cores support RV32 I, M, C, & F ISA and custom ISA extensions for DSP, Bit Manipulation and Vector/SIMD operations
- FC core supports Machine and User (RV32-U) privileged modes, with Physical Memory Protection (PMP)
- Transprecision floating-point support for IEEE 32-bit and 16-bit and bloat16
- Hardware support for 16- and 8-bit fixed point and 16-bit floating point vectors
- µDMA streamed data encrypting and timestamping
- Programmable voltage regulator
- 1 real-time clock, 4 main programmable clocks
- 2 general purpose timer units, each usable as 2 × 32-bit or 1 × 64-bit counters
- On-the-fly hardware AES128/256 encryption / decryption
- Secured execution support with Memory Protection Unit
- PUF – TRNG
- 1.5MB interleaved + 64kB non-interleaved retentive L2 memory
- 2MB internal NV eMRAM
- Optional external high-speed low power SDRAM and Flash with mappable virtual memory support
- Execute in place (XIP) support
- 32kHz external quartz
- BGA 5.5mm × 5.5mm (TBC) and WLCSP 3.72mm × 3.76mm package options

1.5 I/O Interfaces

- 2-lane MIPI CSI-2
- 8-bit Camera Parallel Interface (CPI)
- 4 × SPI Master/Slave
- 4 × I2C Master/Slave & 1 × I3C Master
- 3 × SAI/I2S full duplex PDM/PCM with up to 16 TDM channels per SAI
- 5 × U(S)ART with hardware flow control
- 2 × switchable HyperBus, Quad-SPI, OctoSPI, SDIO memory interfaces
- GPIO / PWM / JTAG

1.6 Tools

- RISC-V C/C++ toolchain with added optimizations based on GNU toolset (GCC & GDB)
- GAP AutoTiler code generator for explicit memory movement
- Generator library with DSP/NN kernel support
- GAP*Flow* toolset providing end-to-end code generation from NN frameworks
- PULP OS, FreeRTOS™ OS support
- Cross OS PMSIS cluster & device API
- Debug support including on-chip debug
- GVSOC SoC Simulator and visual code profiler

1.7 Libraries

- Parallelized, vectorized and highly optimized software components:
  - Data Analysis (FFT, MFCC, etc.)
  - Deep Learning (CNN/DNN based)
  - Image Processing (HOG, DOG, Viola-Jones, etc.)

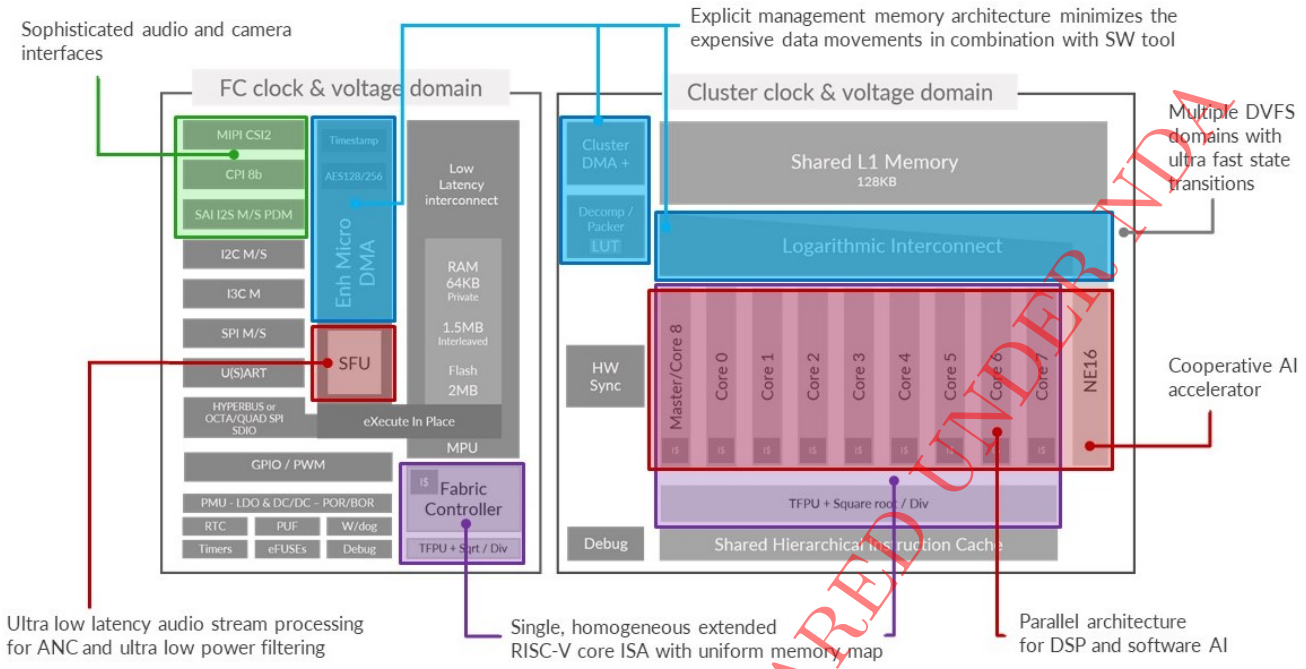
Chapter 2

Introduction to the GAP9 IoT Application Processor

Rich signal processing has been revolutionized by new deep neural network approaches. These have consistently outperformed classical algorithms for classification, detection and transformation algorithms on input sources such as sounds, images, radar, biosignals and more. There is a significant amount of interest in bringing these approaches to highly energy constrained edge devices. These include IoT sensors such as image-based building sensors, wearable devices such as wristbands or smart glasses and hearable products such as earbuds and headsets. GAP9 processor focuses on bringing significant capabilities to extremely power constrained applications, with a unique combination of simple to use, flexible, ultra-low power and latency processing capability. Those features are introduced in this chapter.

2.1 The Building Blocks

GAP9 is made of 3 fundamental building blocks, an MCU type controller that we call the *Fabric Controller*, a smart peripheral controller (the µDMA) integrating a sample by sample audio Smart Filtering Unit (SFU) and a parallel compute engine that we call the *Cluster*. The Fabric Controller (FC) is responsible for managing peripheral devices and controlling application execution on GAP9. The µDMA allows autonomous, low energy peripheral management and on-the-fly calculation through specialized processing blocks such as the SFU for ultra-low latency tasks such as Adaptive Noise Concellation. The cluster provides a flexible, on demand, high performance programmable calculator for any task that demands significant compute resources such as Digital Signal Processing or Machine Learning algorithms.



GAP9 makes extensive use of Dynamic Frequency and Voltage Scaling (DVFS) in multiple different zones (known as *Domains*) of the chip. This allows elements of the chip to be entirely switched off when not in use but also the actual capabilities and energy consumption to be precisely tuned to the requirements of the task being executed.

When a voltage domain is active GAP9 uses automatic clock gating to stop clocking smaller functional blocks when they are not in use further reducing power consumption.

GAP9 has a rich set of peripheral interfaces including a 2-lane CSI-2 interface, parallel camera interface and 3 Serial Audio Interfaces capable of handling up to 16 TDM channels and incorporating 3 input and 1 output PDM channels per interface.

Finally, GAP9 also comes with an SDK, which follows two fundamental philosophies:

- Use tools that are familiar to the targeted developer
- Avoid black boxes that the developer cannot properly debug

2.2 The Fabric Controller

The FC contains a single core that is responsible for coordinating the activity on GAP9. All GAP9's cores in the FC and cluster are based on the RISC-V ISA extended with custom instructions for operations such as MACs, zero-cycle loops, saturation and clipping operations, bit level arithmetic and lightweight 8-bit and 16-bit vector instructions. The use of a single core design simplifies development – only a single compiler toolchain is required.

All the cores in GAP9 include a Transprecision Floating Point Unit handling IEEE 32-bit, 16-bit and BFloat16 floating point numbers. The vector unit in the cores can handle vectors of IEEE-16 or BFloat16 operands.

The FC area of the chip also contains the main memory made up of 1.5MB of RAM, which can optionally be retained in blocks when in low power modes and a 2MB area of non-volatile memory using state-of-the-art eMRAM technology. eMRAM, as opposed to Flash memory, provides high speed read and write access, which gets close to the performance of RAM. It's ideal for storing filter coefficients for neural network tasks. It also can be used to store firmware enabling reduced system cost when no external memory is necessary.

GAP9 has two[1] external memory interfaces that can be individually configured to support Octo- or Quad-SPI, HyperBus or SD memories. The total bandwidth of each external memory interface is 370MB/sec. GAP9 incorporates a virtual memory mapping capability that allows both external and internal memories to be mapped into

[1] Only 1 is available in WLCSP package

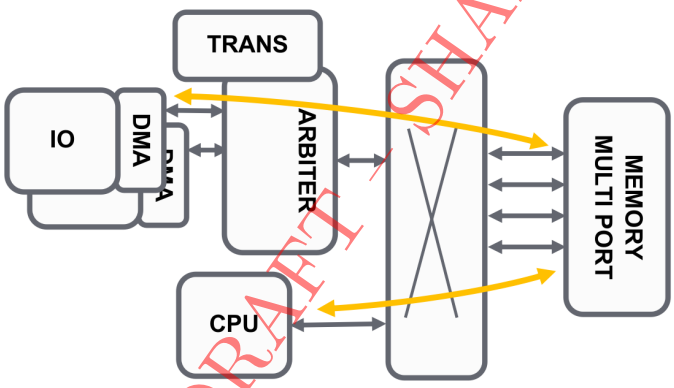
GAP9's memory space with caching in the L2 area. This allows execute-in-place execution of code contained in external memory.

All of the power supplies and oscillators necessary for GAP9 are controlled by the FC via an integrated power management system. The FC can enter a deep sleep state consuming a few tens of µA of current from which it can wake up in a few milliseconds.

2.3 The µDMA

The FC talks to peripherals via an intelligent peripheral controller that we call Micro-DMA. This differs from a classic DMA unit in that it is not executing transactions over a shared system bus. The µDMA has dedicated channels to the L2 memory and a series of dedicated DMA channels, individually connected to and controlled by peripherals. This reduces contention between peripheral dataflow and internal activity improving performance and reducing power consumption.

The µDMA is capable of autonomously executing complex transactions with an ability to process information passing through it on-the-fly. This processing capability ranges from timestamping information arriving from multiple interfaces and on-the-fly AES 128/256 encryption and decryption to complex, ultra-low latency sample by sample audio filtering that can be used to implement state-of-the-art Active Noise Cancellation. We call this audio filtering unit the *Smart Filtering Unit* and will explain it in more detail later in this chapter.



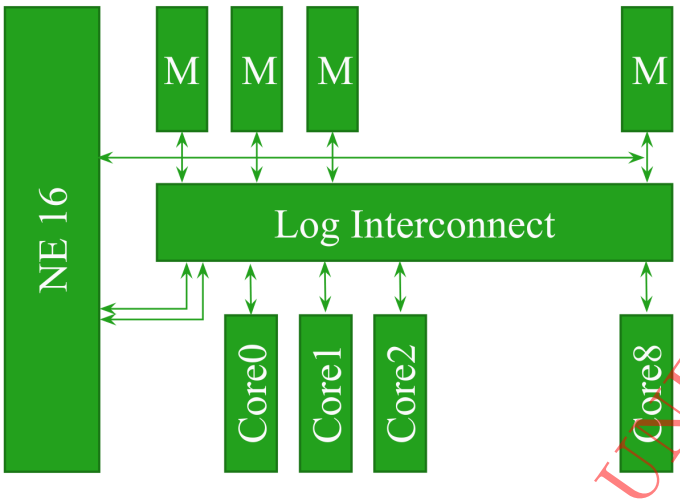
The µDMA reduces the power consumption of data acquisition and transmission by reducing the need for (or entirely eliminating) intervention by the processing cores in GAP9. It also enables ultra-low latency on-the-fly data processing even directly between two interfaces.

The MicroDMA is connected to a rich set of interfaces including ones specialized for sound sources and sinks (SAI) and cameras (MIPI CSI-2 & CPI[2]) and general purpose interfaces such as SPI, UART, I2C & I3C and GPIO.

2.4 The Cluster

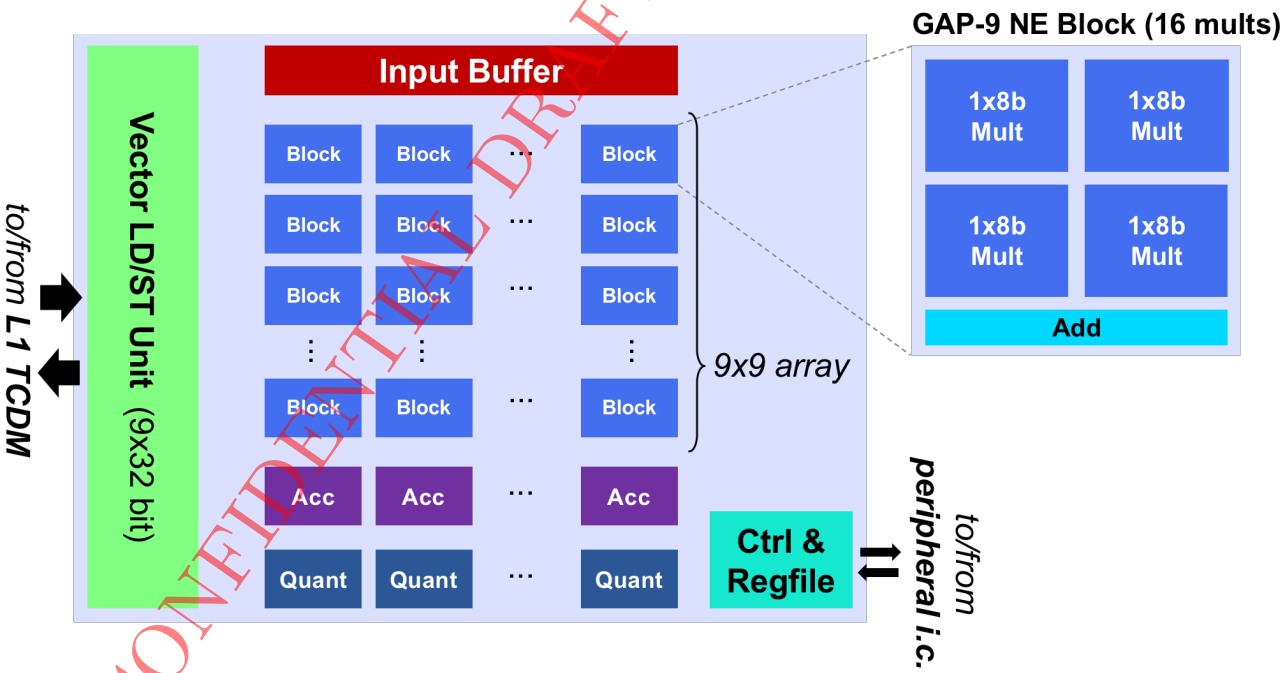
GAP9's cluster includes 9 RISC-V cores (identical to the FC core) coupled with a shared L1 memory area. GAP9's cluster efficiently exploits the parallel nature of most DSP and Machine Learning algorithms to allow them to be run at a lower clock rate for the same performance. This, in turn, allows GAP9's core voltage to be reduced bringing a quadratic reduction in energy consumption. The cores all have their own program counter and can run different code giving a large choice in parallelization strategies.

[2] CPI interface is not available in WLCSP pakage.



To efficiently implement parallel algorithms all the synchronization primitives such as forks, joins & semaphores are implemented in hardware. This allows a task to be forked to all cores in under 5 cycles. When individual cores in a group complete a task they enter a wait state in one cycle where they are immediately clock gated. When all the cores complete in one cycle they all start running again. This allows parallelism to be exploited in algorithms that require frequent synchronization between different threads.

GAP9's cluster architecture allows dedicated hardware accelerators to be combined with the cores. GAP9 includes a dedicated accelerator, Neural Engine 16 (NE16), for the streamed multiply/accumulate (MAC) operations that characterise neural networks. NE16 has the same shared access to the L1 memory as the cores and communicates with them via a sophisticated event based controller. This allows the energy and speed benefits of a dedicated hardware block to benefit from the flexibility of general purpose cores. NE16's design can be focused on delivering the best energy performance for the most critical operations without sacrificing functionality filled in by the cluster cores.



NE16 is however highly flexible, it can handle CNN, RNN or vector/matrix multiplications with 16- or 8-bit features and from 8 to 2 bit weights. It natively supports asymmetric, scaled quantization. The varying precision can be allocated to individual layers. If float execution is needed for some layers these can be executed by software kernels on the cluster cores with vectorized IEEE16 or BFloat16 features and weights. Our kernel library includes a wide range of neural network and signal processing algorithms fully optimised for the cluster, ready for use.

This tightly coupled combination of a programmable multi-core compute cluster and dedicated hardware accelerator

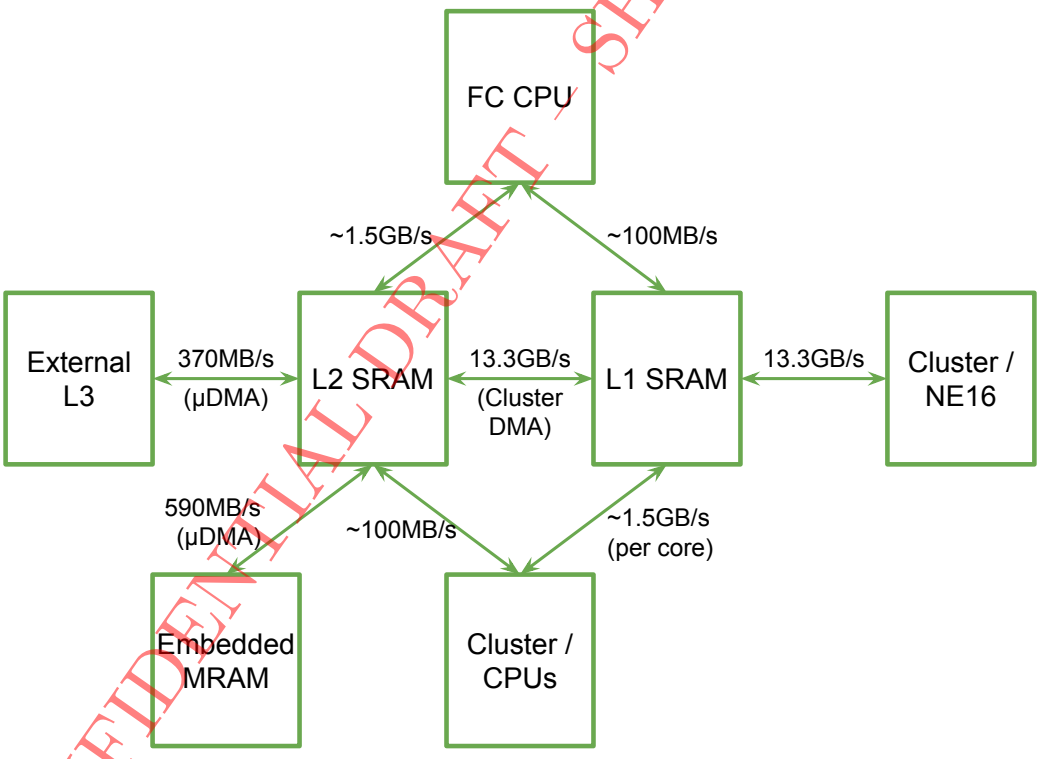
is unique in the market and is flexible enough to adapt to the latest, state-of-the-art signal processing and AI techniques. We have found out the while it is possible to produce architectures that give greater efficiency on some synthetic example models, in real life applications the cluster's flexibility provides the optimal balance of network accuracy and energy performance.

2.5 Explicit Memory Movement

Instruction and data movement incurs a large energy cost. GAP9's cluster incorporates a hierarchical instruction cache that ensures that a sequence of instructions used by more than one core is only loaded once; however GAP9 incorporates no data caching.

All the cores in GAP9 see exactly the same memory map; however to improve performance, data movement should be hidden behind processing as much as possible. Classically this is achieved through some form of data cache. However for GAP9's workload, essentially streams of images, sounds, and so on whose dimensions are already known at compile time a data cache would be highly inefficient. The expected cache hit rate, how often loaded data would actually be in the cache, would be in the order of 30%. This would result in stalls and misloads causing loaded data to be thrown away – clearly a useless energy cost. In GAP9 sophisticated Direct Memory Access (DMA) units are used to explicitly move data.

While this cuts down on energy consumption, the problem of correctly positioning data elements for an operation or sequence of operations and properly scheduling data movement between external L3 and internal L1 and L2 memory areas to bring data as close as possible to the processing element using it is difficult to solve manually.



We resolve this problem using a sophisticated optimisation and code generation tool called the GAP AutoTiler. The AutoTiler takes as input a series of models of a computation graph (this could be a neural network graph for example) and the kernels that implement its operations and configurable memory constraints and discovers a solution for parameter placement and movement that minimizes the amount of times a piece of data is moved while ensuring that it is in the cluster L1 when it is needed.

The AutoTiler is one in a series of tools that makes up the GAP SDK, available from GreenwavesTechnologies GitHub repository:

- RISC-V GCC compiler with native support of GAP9 ISA extensions.

- GVSOC simulator, that allows code to be run on the developers workstation. Full visibility of the activity inside the simulated GAP9 is provided through signal traces in the GAP profiling tool. Many peripherals are simulated by GVSOC allowing complete applications to be run.
- NNTOOL translator that can take standard TensorFlow Lite or ONNX graph descriptions and produce highly optimized C code that executes the graph using the extensive GAP kernel library. NNTOOL takes the graph as input and produces a model for the AutoTiler tool that generates the C code.

Contact GreenWaves Technologies for more details on these different tools.

2.6 Processing Audio Streams

While the GAP9 cluster covers the frequency domain filtering, physics modelling and neural network applications that are required for advanced audio applications, sample by sample filtering at ultra-low latency requires dedicated hardware.

GAP9 Smart Filtering Unit (SFU) provides a flexible block that can implement sample by sample filtering on audio streams coming from and going to PDM Audio interfaces, peripheral interfaces (including SAI, SPI, I2C) and L2 memory. Highly configurable, dedicated hardware implements a wide range of filtering types along with splitters, mixers, limiters and so on. The SFU integrates multi-channel / multi-tracker asynchronous sample rate conversion and flexible PDM modulator and demodulators. All these blocks can be combined into entirely user-defined multiple audio filtering graphs which can be run simultaneously. Furthermore, filter coefficients can be dynamically updated without stopping the graph and individual graphs can be loaded and unloaded.

The SFU's internal precision can be set to two levels: 32-bit input + 32-bit state into 64 bits or 32-bit input + 64-bit state into 96 bits. This is considerably more precision than most DSPs, which simplifies filter design. Also, since the SFU is a sub-system that you configure and then run, there is no instruction load penalty since there are no instructions to load. This has both a performance and energy benefit.

There are two primary scenarios for using the SFU: as a continuous filter graph between two PDM interfaces for applications such as ANC and as a filtering coprocessor for the FC and Cluster between blocks of samples in L2 for equalization and other sound effects.

The SFU enables real time operation on individual samples at a 768kHz sample rate, a structural latency of 1.35µs. Since it sits in its own DVFS domain, its performance and energy consumption can be precisely tuned to the task it is executing.

GAP9 incorporates 3 sophisticated SAI interfaces supporting up to 16 TDM inward, outward or forwarded slots on each interface. Each interface also incorporates two 2-channel PDM interfaces selectable as input or output. Each channel can be put into a clockless differential PDM mode which can be directly connected to an analog, minimal latency sigma-delta amplifier.

The GAP SDK includes GreenWaves Technologies's AudioTools framework providing a familiar interface via Mathworks plugins for filter design and a GraphTool utility allowing integration of SFU filters, Cluster Filtering and Neural Networks and graph elements running on the FC.

2.7 Security

Security is an important consideration for any edge device. GAP9 incorporates several features to ensure that firmware is not tampered with and that confidential algorithms and neural network parameters are protected.

GAP9 incorporates a bank of one time programmable eFuses that can be used to store keys and to enable and disable firmware features. GAP9 protects system memory with a memory protection unit and hardware machine and user level privilege modes. It also integrates a physically unclonable function (PUF) hardware block that can be used to generate a unique chip ID and seed a strong random number generator for cryptographic functions. Finally the µDMA incorporates a hardware AES-128/256 encryption and decryption engine.

These hardware features enable the implementation of secure boot of encrypted firmware, on-the-fly decryption of network parameters fetched from external memory and compartmentalized application code execution.

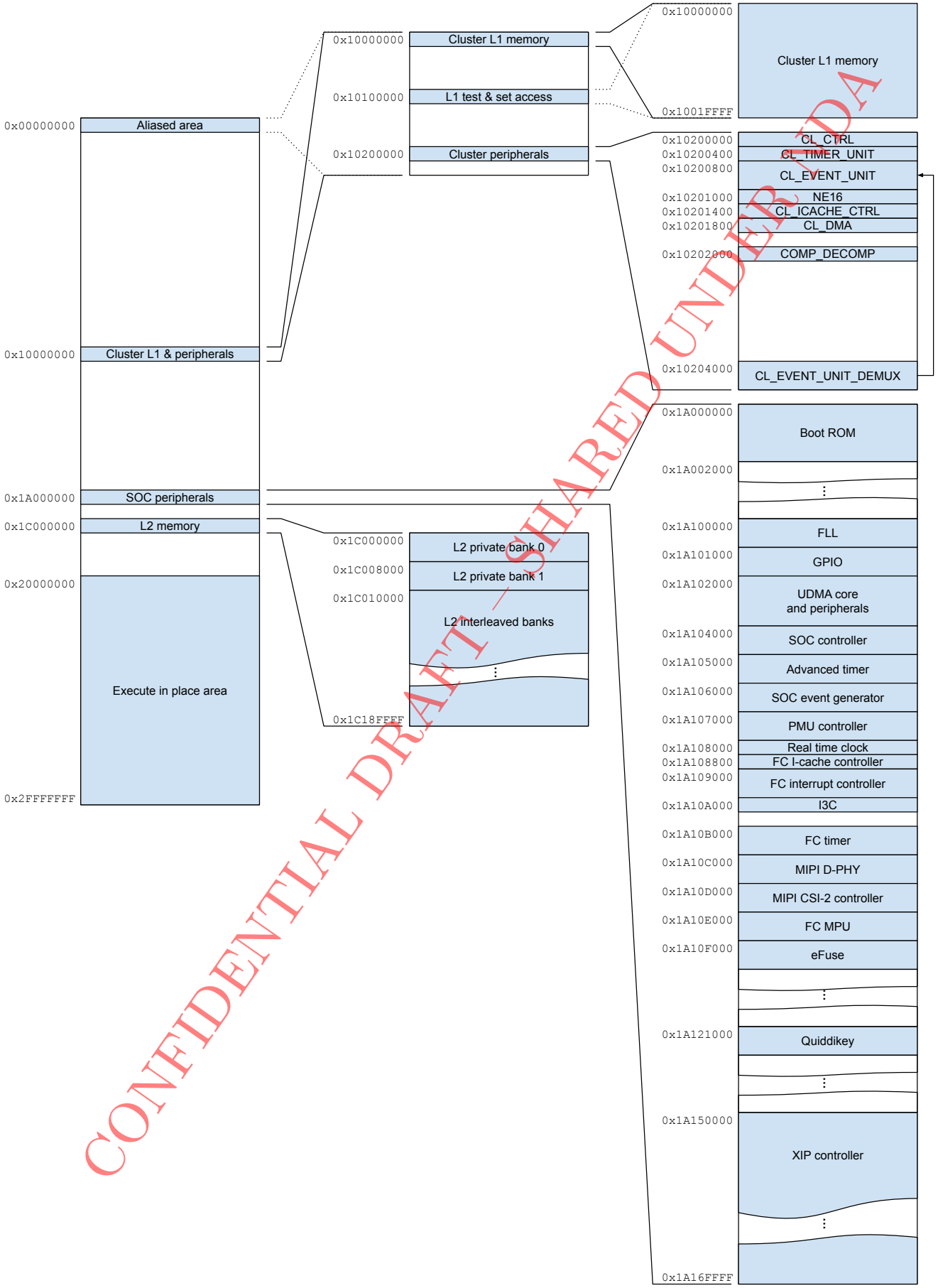
Chapter 3

Memory Map

3.1 Overview

GAP9 address map organization is shown on the next picture. It is split into 3 main parts:

- Address space for internal memory and peripherals, starting at 0x1000000, which is split into cluster address space (L1 access and cluster peripherals) and SoC space (boot ROM, L2 memory and SoC domain peripherals).
- An aliased area for cluster space at address 0x00000000, which allows to access L1 directly from address 0.
- Address space used by XIP for execute-in-place feature (256MB).



3.2 Details

IP instance	IP instance description	Address	Aliased address	Length
ALIASED_AREA	Aliased access to cluster address map	0x00000000		0x400000
CL_TCDM	Cluster L1 RAM (64kB)	0x10000000	0x00000000	0x20000
CL_TEST&SET	Test & Set Access To Cluster L1	0x10100000	0x00100000	0x20000
CL_CTRL	Cluster Controller	0x10200000	0x00200000	0x400
CL_TIMER_UNIT	Cluster Timer	0x10200400	0x00200400	0x400
CL_EVENT_UNIT	Cluster Event Unit	0x10200800	0x00200800	0x800
NE16	NE16 Neural Accelerator	0x10201000	0x00201000	0x400
CL_ICACHE_CTRL	Cluster Instruction Cache Controller	0x10201400	0x00201400	0x400
CL_DMA	Cluster DMA	0x10201800	0x00201800	0x400
COMP_DECOMP	Compressor/Decompressor	0x10202000	0x00202000	0x400
CL_EVENT_UNIT_DEMUX	Cluster Event Unit (Cluster Cores Private)	0x10204000	0x00204000	0x800
ROM	ROM (8kB)	0x1A000000		0x2000
FLL	FLL Clock Generator	0x1A100000		0x1000
GPIO	GPIO Controller	0x1A101000		0x1000
UDMA_CTRL	uDMA Controller	0x1A102000		0x80
SPI0	SPI Interface 0	0x1A102080		0x80
SPI1	SPI Interface 1	0x1A102100		0x80
SPI2	SPI Interface 2	0x1A102180		0x80
SPI3	SPI Interface 3	0x1A102200		0x80
UART0	UART Interface 0	0x1A102280		0x80
UART1	UART Interface 1	0x1A102300		0x80
UART2	UART Interface 2	0x1A102380		0x80
UART3	UART Interface 3	0x1A102400		0x80
UART4	UART Interface 4	0x1A102480		0x80
I2C0	I2C Interface 0	0x1A102500		0x80
I2C1	I2C Interface 1	0x1A102580		0x80
I2C2	I2C Interface 2	0x1A102600		0x80
I2C3	I2C Interface 3	0x1A102680		0x80
MEM0	Memory Interface 0	0x1A102700		0x80
MEM1	Memory Interface 1	0x1A102780		0x80
RESERVED	Reserved	0x1A102800		0x80
SAI0	Serial Audio Interface 0	0x1A102880		0x80
SAI1	Serial Audio Interface 1	0x1A102900		0x80
SAI2	Serial Audio Interface 2	0x1A102980		0x80
CPI	Camera parallel interface	0x1A102A00		0x80
CSI-2	CSI-2/uDMA interface	0x1A102A80		0x80
MRAM	eMRAM Controller	0x1A102B00		0x80
TIMESTAMP	Timestamp unit	0x1A102C00		0x80
DUAL_CORE_AES	Dual-core AES encryption engine	0x1A102C80		0x80
AES	AES encryption engine	0x1A102D00		0x80
SFU	Smart Filtering Unit	0x1A102D80		0x80
FFC	Fixed-/Floating-point Converter	0x1A102E00		0x80
LIN_ADDRGEN0	Linear Address Generator	0x1A103000		0x20
LIN_ADDRGEN1	Linear Address Generator	0x1A103020		0x20
...	...	...		...
LIN_ADDRGEN63	Linear Address Generator	0x1A1037E0		0x20

IP instance	IP instance description	Address	Aliased address	Length
2D_ADDRGEN0	2D Address Generator	0x1A103800		0x20
2D_ADDRGEN1	2D Address Generator	0x1A103820		0x20
...	...	...		...
2D_ADDRGEN7	2D Address Generator	0x1A1038E0		0x20
FIFO_ADDRGEN0	Channel-to-Channel FIFO	0x1A103900		0x20
FIFO_ADDRGEN1	Channel-to-Channel FIFO	0x1A103920		0x20
...	...	...		...
FIFO_ADDRGEN7	Channel-to-Channel FIFO	0x1A1039E0		0x20
SOC_CTRL	SoC Controller	0x1A104000		0x1000
ADV_TIMER	Advanced PWM Timer	0x1A105000		0x1000
SOC_EVENT_GENERATOR	SoC Event Generator	0x1A106000		0x1000
PMU	PMU Controller	0x1A107000		0x1000
RTC	Real Time Clock	0x1A108000		0x800
FC_ICACHE_CTRL	FC Instruction Cache Controller	0x1A108800		0x800
FC_ITC	FC Interrupt Controller	0x1A109000		0x1000
I3C0	I3C Interface 0	0x1A10A000		0x800
FC_TIMER_UNIT	FC Timer	0x1A10B000		0x1000
CSI2_DPHY	MIPI D-PHY for CSI-2	0x1A10C000		0x1000
CSI2_CTRL	MIPI CSI-2 Controller	0x1A10D000		0x1000
MPU	Memory Protection Unit	0x1A10E000		0x1000
EFUSE	eFuse	0x1A10F000		0x1000
QUIDDIKEY	Quiddikey PUF	0x1A121000		0x1000
XIP_CTRL	Execute In Place Controller	0x1A150000		0x20000
L2_PRIV0	L2 Private RAM 0 (32kB, 2 banks)	0x1C000000		0x8000
L2_PRIV1	L2 Private RAM 1 (32kB, 2 banks)	0x1C008000		0x8000
L2_INTERL	L2 Interleaved RAM (1.5MB, 12 banks)	0x1C010000		0x180000
XIP	Execute In Place Space	0x20000000		0x10000000

Chapter 4

GAP9 Architecture

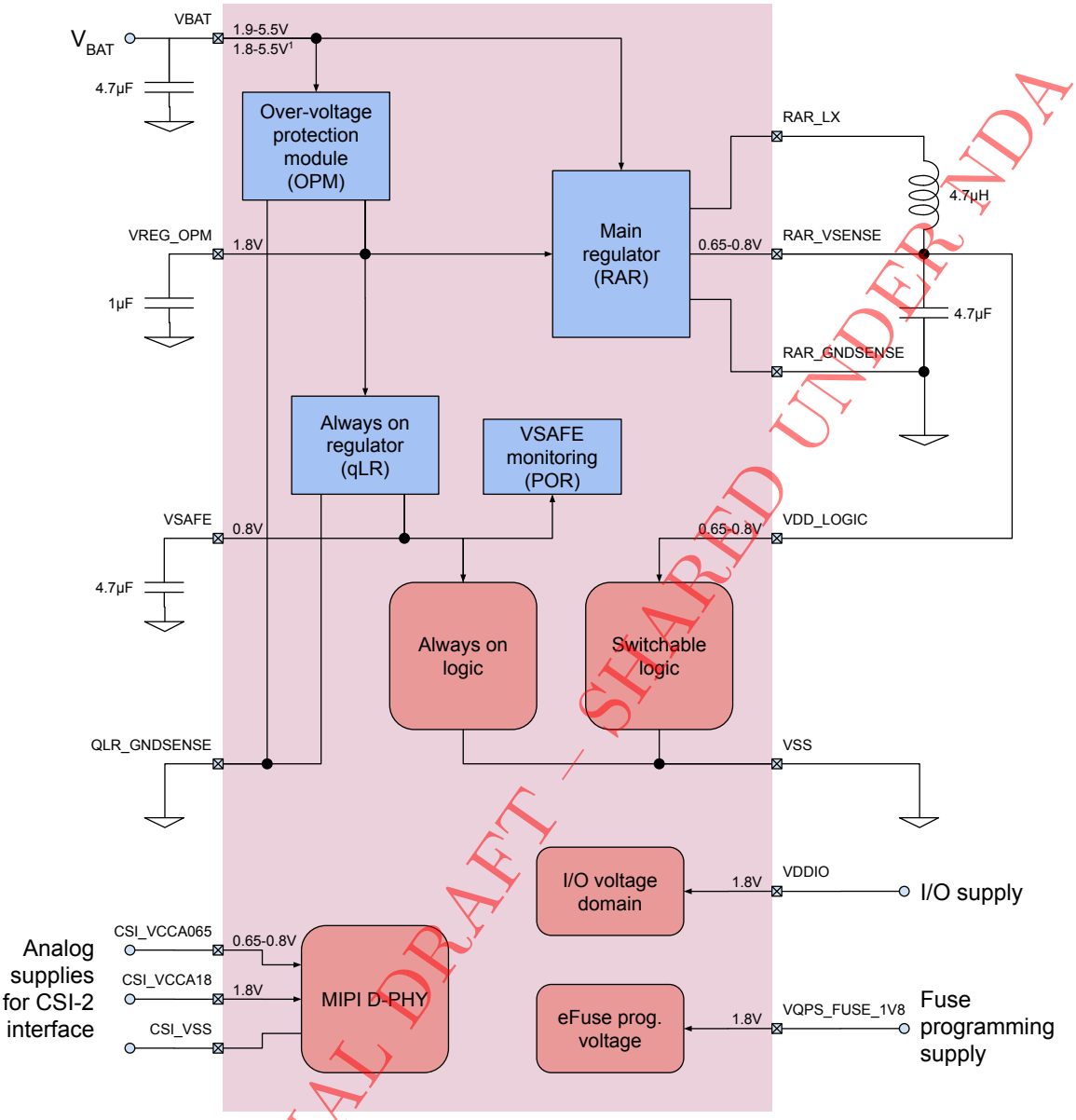
4.1 Power

4.1.1 Power Supplies

4.1.1.1 Overview

The figure below presents an overview of GAP9 power supplies. `VBAT` is the main power supply for the chip, and is used to generate internal power supplies for the logic. Supported range for `VBAT` is 1.9 to 5.5V[1].

[1] The functionality of GAP9 with `VBAT` = 1.8V has also been assessed, allowing for a 1.8V common-rail supply for `VBAT` and `VDDIO`. This comes with some restrictions on the allowed temperature range – see Operating conditions for more details.



4.1.1.2 Supplies

An always-on over-voltage protection module (OPM) is used to generate an internal 1.8V supply required to control other internal regulators. It features local supply monitoring which allows to properly reset those regulators in case of important voltage droops (see next section).

Note: The OPM regulator purpose is to supply internal power-related blocks only. The 1.8V `VREG_OPM` pad is dedicated to the connection of the required external capacitor and must not be connected to another component or pad.

The power supply of the always-on logic is regulated by a ultra-low quiescent LDO, called qLR, which produces a fixed 0.8V voltage (`VSAFE`). This voltage is internally connected to the always-on power domain, which embeds for example the wake-up mechanisms, the real-time clock, and the registers which retain their values during deep sleep.

Note: The 0.8V `VSAFE` pad is dedicated to the connection of the required external capacitor. It should not be

connected to another component.

The power supply of the other power domains, in the 0.65-0.8V range, is generated by a configurable regulator, called RAR, which is the main regulator of the chip. This regulator embeds a DC-DC converter, used for nominal operations, and an LDO used to supply low loads (<3mA) in low power or retention modes. When the DC-DC converter is in operation, the regulated voltage is obtained from LC filtering of `RAR_LX` output, and fed back into `RAR_VSENSE` for proper regulation. When the LDO converter is used, the regulator output is directly available through `RAR_VSENSE` pad. In both cases, the regulated output must be connected externally to the `VDD_LOGIC` pads to supply power to the internal logic[2].

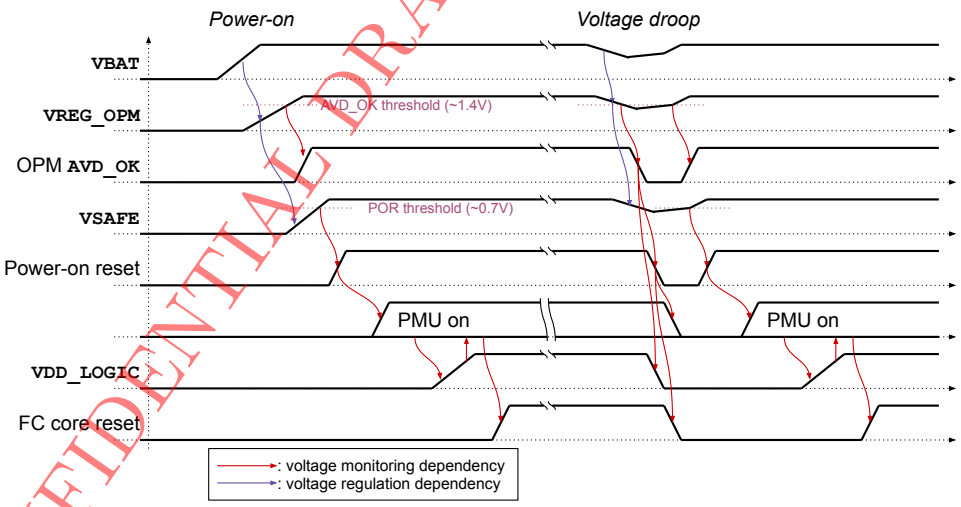
Other power supplies are required to operate GAP9:

- `VDDIO`: 1.8V external[3] power supply for chip I/Os and eMRAM supply.
- CSI-2 MIPI PHY: when CSI-2 interface is used, the MIPI PHY must be supplied through `CSI_VCCA065` (0.65V-0.8V), `CSI_VCCA18` (1.8V[3]) and `CSI_VSS` (ground).
- `VQPS_FUSE_1V8`: 1.8V external[3] power supply for eFuse programming. Can be left floating or grounded if eFuse programming is not required.

4.1.1.3 Local Supply Monitoring

Supply monitoring features are integrated in GAP9 to ensure proper boot of the chip, and a clean reset in case of supply droop, as described below. The next figure illustrates how these features take part in the boot sequence.

When `VBAT` rises to power the chip, the OPM boots and starts its regulation of the `VREG_OPM` supply. The OPM offers a monitoring of its input and output supplies, which keeps the `AVD_OK` signal low until `VREG_OPM` has settled. This signal ensures that 1) the `VSAFE` monitoring block keeps asserting (low) the power-up reset, and 2) the RAR remains in its default state, i.e. switched off. Once the `AVD_OK` rises, `VREG_OPM` is stable enough to ensure a proper behavior of other power-related blocks. The POR monitors `VSAFE` ramping-up, and releases the reset once its level is high and stable enough. The always-on logic can then start executing, in particular the PMU, which manages the rest of the boot process (e.g. starting the DC-DC regulator, switching on power domains). More details on the complete boot process are given in the Boot Process chapter.



When GAP9 is on, the monitoring of OPM input and output supplies is maintained, in order to detect voltage droops, for instance when other parts of the system induce a voltage drop on `VBAT`. When `VREG_OPM` drops below its monitored threshold, the OPM lowers its `AVD_OK` signal, which in turn 1) resets the `VSAFE` monitoring block, which then asserts low the power-on reset, and 2) resets the RAR into its default state, i.e. switched off. This ensures that GAP9 is in a state similar to when `VBAT` rises during cold boot, ready to resume execution when the supplies are back to normal. When a `VBAT` droop is too important to guarantee that the OPM is still able

[2] External connection of `VDD_LOGIC` supply allows to power the chip logic directly by an external regulator in specific cases such as production tests.

[3] `VREG_OPM` cannot be used to provide these supplies.

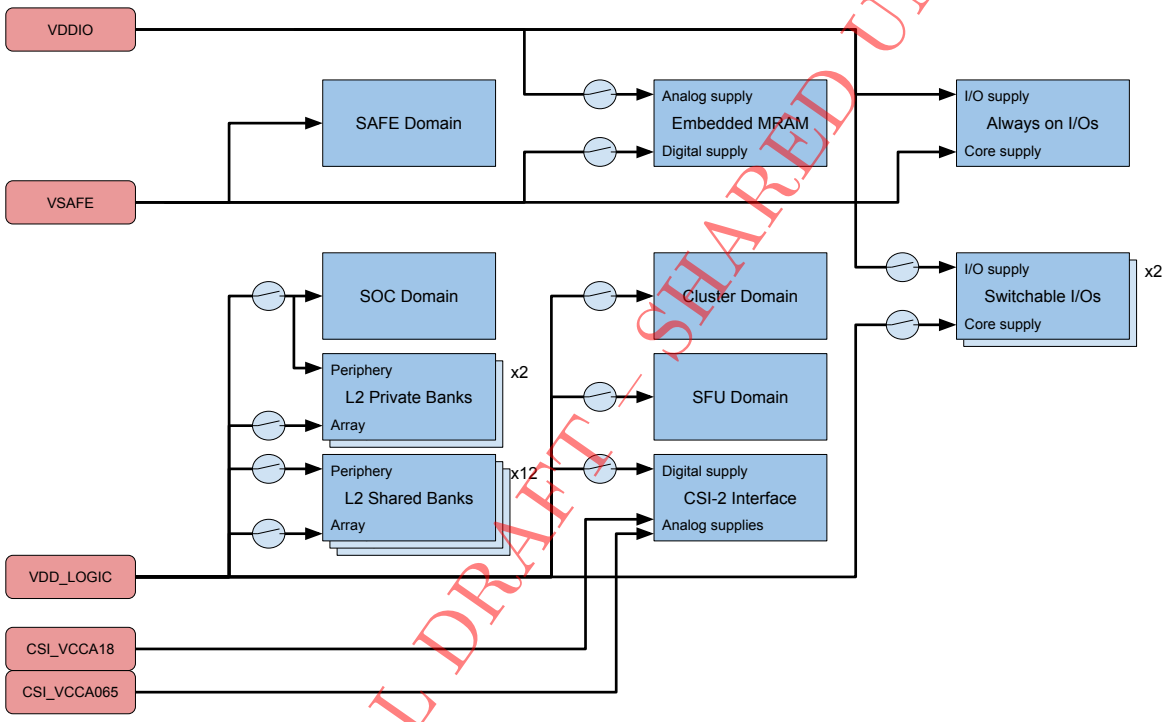
to properly monitor its output, then a safeguard mechanism resets the OPM itself, which also lowers the `AVD_OK` signal, ensuring the same behavior as previously described.

Note: GAP9 does not provide supply monitoring for `VDDIO`.

4.1.2 Power Domains

4.1.2.1 Overview

Next figure shows the different power domains of GAP9. Main power domains are introduced below. Unless otherwise stated, switching on or off a power domain is realized by triggering a PMU sequence.



4.1.2.2 SAFE Domain

This is the power domain of the always-on logic. This domain is always powered (0.8V), even in deep sleep. It embeds the logic which is required at all time: power management unit, RTC, wake-up logic (wake-up interfaces, wake-up timers, etc.), always-on registers, etc.

4.1.2.3 SOC Domain

This is the main functional power domain of GAP9. It includes the Fabric Controller, the uDMA, execute in place unit, all peripheral interface controllers but CSI-2, the PUF, the watchdog, etc. The SOC domain is powered by VDD_LOGIC in the 0.65V-0.8V range, and can be switched off for retention and deep sleep modes.

4.1.2.4 L2 Memory

L2 memory is made of 2 32KB "private" banks and 12 128KB "shared" banks. In order to allow for maximum flexibility in the use of L2 memory banks, they are organized in a specific way from the powering standpoint:

- Under SW control, every L2 memory bank can be turned off if not used, in order to save power. This is done by switching off the memory bank array (the part that stores the data) and periphery[4] (the part which controls access to the memory).
- Under SW control, every bank array can be powered or not during retention mode, in order to precisely select data that should be retained in this mode. This is done by selecting which memory arrays remains powered in this mode.

Those features are controlled by the L2_PWR and L2_CTRL registers.

4.1.2.5 Cluster Domain

This domain embeds the 9 accelerator cores, the NE16, as well as the DMA and the compressor/decompressor for efficient data transfer between the cluster L1 memory and the chip L2 memory. The Cluster domain is powered by VDD_LOGIC in the 0.65V-0.8V range, and can be switched off when it is not used.

4.1.2.6 SFU Domain

The SFU power domain is made of the different building blocks of the Smart Filtering Unit. It is powered by VDD_LOGIC in the 0.65V-0.8V range, and can be switched off when it is not used.

4.1.2.7 CSI-2 Interface Domain

The CSI-2 interface domain gathers a CSI-2 controller and a MIPI PHY. Analog power supplies for the PHY are provided externally (see Supplies). Digital power supplies are provided internally from VDD_LOGIC in the 0.65V-0.8V range, through a dedicated power switch.

Note: The externally provided analog supplies should be turned off when CSI-2 internal digital supply is off.

4.1.2.8 Embedded MRAM

In order to operate, the embedded MRAM internally requires a 0.8V supply and a separate supply in the 1.5-1.8V range, as well as a 0.6V reference voltage. The 0.8V supply is provided from VSAFE through a controllable power switch. Another controllable switch provides a 1.8V supply directly from VDDIO. The reference voltage is provided by the DC-DC of the main embedded regulator, and can be disabled when the MRAM is not used (see RARMODE register). These features allow to control the switching on and off of the MRAM depending on application needs.

4.1.2.9 Switchable I/Os

In order to save power when high speed I/Os are not used in functional mode, and in retention and sleep modes, a power cut feature has been implemented in GAP9. Switchable I/Os are split into 2 segments which can be independently switched by the PMU, either when changing power mode or though a request from SW. The first segment is made of the memory controller I/Os. The second segment includes all other switchable I/Os. Refer to pin description for the list of switchable I/Os.

[4] In the case of the private L2 banks, only the switching of array is controllable by SW, the periphery is directly powered by the SOC domain.

4.1.3 GAP9 Power Modes

4.1.3.1 Predefined Configurations

GAP9 comes with a number of predefined power modes, which are described in the table below. Each of these modes (except for sleep modes) can be adapted under software control, depending on application's needs: powering of cluster, SFU, MRAM; modification of logic power supply voltage, fine control of L2 memory banks powering, powering of switchable I/Os. At power-up, upon completion of the boot sequence, GAP9 is in the NOMINAL_VOLTAGE state, with the DC-DC supply set to 0.8V.

Power mode name	Switchable I/Os	SOC domain	L2 memories	SFU domain	Cluster domain	CSI-2	eMRAM
NOMINAL_VOLTAGE	ON(1)	0.65V-0.8V (DC-DC)	0.65V-0.8V (DC-DC)(2)	OFF(3)	OFF(3)	OFF(3)	OFF(4)
LOW_POWER	ON(1)	0.65V (LDO)	0.65V (LDO)(2)	OFF	OFF	OFF	OFF
LIGHT_SLEEP	ON/OFF(5)	OFF	0.65V (DC-DC)(6)	OFF	OFF	OFF	OFF
RETENTIVE	OFF	OFF	0.65V (LDO)(6)	OFF	OFF	OFF	OFF
DEEPSLEEP	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Notes:
(1): ON by default, can be switched off if not used.
(2): Unused L2 banks can be switched off.
(3): OFF by default, may be powered (from DC-DC) in order to be used.
(4): OFF by default, may be powered (0.8V from VSAFE and 1.8V from VDDIO) in order to be used.
(5): The state of switchable I/Os is unchanged when ingressing/egressing LIGHT_SLEEP mode.
(6): Only the array (data) part of selected L2 banks is powered.

4.1.3.2 Transitions Between Power Modes

Modification of GAP9 power state, as well as transitions between the predefined power states, is achieved by a combination of registers configuration and triggering of PMU predefined sequences. A PMU sequence may be triggered by software, using dedicated runtime routines which start a sequence and monitor its completion. While executing a sequence, the PMU automatically controls the different hardware power features involved (regulators, power switches, ABB controller, etc.).

In order to be triggered, a sequence must be mapped onto a PMU interrupt (0-14) using a dedicated runtime routine which accesses PMU internal registers. Sequences mapped onto the four first interrupts (0-3) may also be triggered by wake-up events depending on the SLEEP_CTRL configuration (see WAKEUP_CFG field). This allows to select, through software configuration, in which state GAP9 will be after wake up.

The control of the power state of the L2 banks (extinction and retention) can be done directly by software using the L2_PWR and L2_CTRL configuration registers.

The table below shows the predefined GAP9 sequences. Each cell gives the status of the regulator and power domains at the end of the sequence. An empty cell means that the regulator or power domain state is not affected by the PMU sequence.

Seq ID	Purpose	Regulator	Memory I/Os	Other sw. I/Os	SOC	SFU	CLUSTER	CSI-2	EMRAM
0	Wake up SOC	0.65V-0.8V	ON	ON	ON	OFF	OFF	OFF	OFF
1	Wake up SOC +MRAM	0.65V-0.8V	ON	ON	ON	OFF	OFF	OFF	ON

Seq ID	Purpose	Regulator	Memory I/Os	Other sw. I/Os	SOC	SFU	CLUSTER	CSI-2	EMRAM
2	Wake up SOC +Cluster	0.65V-0.8V	ON	ON	ON	OFF	ON	OFF	OFF
3	Wake up SOC +Cluster +MRAM	0.65V-0.8V	ON	ON	ON	OFF	ON	OFF	ON
4	Go to DEEP-SLEEP	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
5	Go to RETEN-TIVE sleep	0.65V (LDO)	OFF	OFF	RET.	OFF	OFF	OFF	OFF
6	Power off SFU					OFF			
7	Power on SFU	0.65V-0.8V				ON			
8	Power off eM-RAM								OFF
9	Power on eM-RAMR	0.65V-0.8V							ON
10	Power off CSI-2 interface							OFF	
11	Power on CSI-2 interface	0.65V-0.8V						ON	
12	Power off clus-ter						OFF		
13	Power on clus-ter	0.65V-0.8V					ON		
14	Power off I/Os (not mem. I/Fs)			OFF					
15	Power on I/Os (not mem. I/Fs)			ON					
16	Power off I/Os (memory I/Fs)		OFF						
17	Power on I/Os (memory I/Fs)		ON						
18	Go to LIGHT_-SLEEP	0.65V			RET.	OFF	OFF	OFF	OFF
19	Wake up don't touch I/Os	0.65V-0.8V			ON	OFF	OFF	OFF	OFF
20	Force ABB setting SOC @0.65V	0.65V-0.8V			ON				
21	Force ABB set-ting SOC+SFU @O.65V	0.65V-0.8V			ON				
22	Force ABB setting cluster @0.65V	0.65V-0.8V					ON		
23	Force ABB setting SOC @0.8V	0.65V-0.8V			ON				

Seq ID	Purpose	Regulator	Memory I/Os	Other sw. I/Os	SOC	SFU	CLUSTER	CSI-2	EMRAM
24	Force ABB setting SOC+SFU @0.8V	0.65V-0.8V			ON				
25	Force ABB setting cluster @0.8V	0.65V-0.8V					ON		
26	Wake up SOC +SFU	0.65V-0.8V	ON	ON	ON	ON	OFF	OFF	OFF
27	Wake up SOC +SFU+MRAM	0.65V-0.8V	ON	ON	ON	ON	OFF	OFF	ON
28	Wake up SOC +SFU+Cluster	0.65V-0.8V	ON	ON	ON	ON	ON	OFF	OFF
29	Wake up SOC +SFU+Cluster +MRAM	0.65V-0.8V	ON	ON	ON	ON	ON	OFF	ON

Note: PMU start-up sequence at power-up is equivalent to sequence 0 in the table above with DC-DC voltage set to 0.8V, meaning that the chip boots at 0.8V, with I/Os enabled.

Note: The PMU start-up and wake-up sequences also wait for the settlement of external VDDIO power supply before enabling internal power domains. However there is no precise monitoring of VDDIO level in GAP9. Detection of VDDIO browning out, which could impact the proper operation of GAP9, should be done at system level, and for example trigger a chip reset through its dedicated pin.

4.1.4 Wake-Up from Sleep Modes

Before entering a sleep mode, the software has to configure the wake up sources which may bring it back to an active mode. This is done using a number of configuration registers from the SoC Controller device. The SLEEP_CTRL register in particular allows to enable wake-up sources, and to identify the actual source after wake-up.

The WAKEUP_CFG field of the SLEEP_CTRL register allows to select the PMU *interrupt* (not sequence) to be triggered at wake-up, from 0 to 3. These interrupts are mapped by default onto the predefined sequences 0 to 3 of this table, but can be remapped onto different sequences by reconfiguring PMU internal registers.

The following paragraphs detail the different wake-up mechanisms.

Note: The clocks mentioned in this section are described later in this document (see the System Clocks section).

4.1.4.1 Counter-Based Wake-Up

This feature, controlled by the SLEEP_CNT_CTRL register, allows to wake GAP9 up after a determined amount of time thanks to a counter. When the counter reaches the expected value, the wake-up sequence is triggered.

The time reference for the counter is the SOC slow clock, i.e. either the REF FAST clock divided according to CLK_DIV_REF_FAST register, or the 32kHz REF SLOW clock if present. Selection is done through the REF_CLK_MUX register.

4.1.4.2 RTC Wake-Up

When the 32kHz REF SLOW clock is available, it can be configured to trigger interrupts at precise times over long ranges. The SLEEP_CTRL register can be use to enable RTC interrupt as a wake-up source.

4.1.4.3 Wake-Up Interface

The wakeup interface uses 4 pads to act as a SPI-slave-like interface: WAKEUP_SPI2_CS0, WAKEUP_SPI2_SCK, WAKEUP_SPI2_SDI (MISO), WAKEUP_SPI2_SDO (MOSI) (see pin description).

The wake-up interface can be activated using the MUX field of the SLEEP_SPIS_CTRL register. Note that if it is disabled, the WAKEUP_SPI2_CS0 and WAKEUP_SPI2_SDO pads can be used as GPIO wake-up sources, as detailed in the next section.

Before entering sleep, the software must set the SPIS_SW_DONE field of the SLEEP_SPIS_CTRL register to 0. EXT_WAKEUP_EN field of the SLEEP_CTRL register must also be set to enable the wake-up. When the wake-up interface is activated, it takes the priority to access the 4 involved GAP9 pads until the wake-up process is completed.

During the sleep, when an external SPI master device sends the 0x3F command through SPI, it is decoded as a request to wake-up GAP9 and triggers the wake-up sequence. The SPI master device should then regularly send 0x05 commands to read the wake-up status. This status will be 0 until the SPIS_SW_DONE field of the SLEEP_SPIS_CTRL is set to 1 by GAP9 software after wake-up, signaling the end of the boot process. The SPI master device will then complete this handshake protocol by sending a 0x71 command. Upon reception of this command, the wake-up interface will immediately release GAP9 pads to their nominal behavior – for example the default SPI2 slave interface, to go on with SPI communication with the external device.

4.1.4.4 Wake-Up on External Events

GAP9 offers 16 GPIO-based wake-up sources (12 for WLCSP package due to the limited number of available pads). See the pin description for the list of physical pads that can be used as GPIO wake-up sources. Each source can be configured to trigger on high or low value, using the SLEEP_GPIO_CTRL, and the wake-up can be activated through the EXT_WAKEUP_EN field of the SLEEP_CTRL register. The events triggered during GAP9 sleep are registered in the EXT_EVENT field, which is cleared when EXT_WAKEUP_EN is cleared.

4.1.4.5 IDLE Mode

The IDLE mode is used to gate GAP9 SOC and PERIPH clocks when no activity is to occur during short periods of time, in order to save some power. This is not exactly a sleep mode, because it does not change GAP9 power state (regulators and switches keep their respective states). This limits the possible power savings but at the same time reduces to a minimum the overhead to enter/leave this mode.

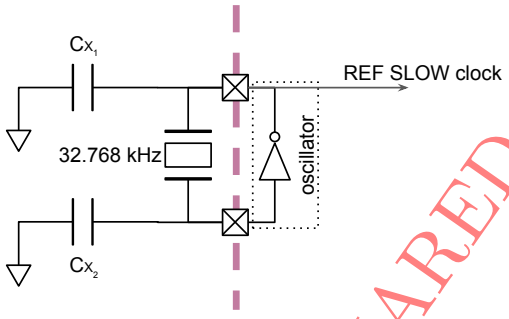
The IDLE mode is triggered by writing a value into the REF_COUNTER field of the IDLE_MODE register. A hardware mechanism ensures that any pending XIP transaction is completed, then the clocks are gated, freezing all activity on FC, L2, uDMA and peripherals. Activity will resume automatically when the configured number of cycles has occured. The time reference for the counter is the SOC slow clock, which is selected from either the REF FAST clock divided according to the CLK_DIV_REF_FAST register, or the 32kHz REF SLOW clock if present. Selection is done with the REF_CLK_MUX register.

4.2 Clocking

4.2.1 Low-Speed Reference Clock

GAP9 offers a low-speed 32.768kHz reference clock, which is used for real time clock, watchdog timer, always-on timer, as well as a reference signal for PWM signal generator event units. This clock is referred to as the *REF SLOW clock* in the following.

The REF SLOW clock is generated by a low power oscillator, from an external crystal connected to the `SLOW_XTAL_XA` and `SLOW_XTAL_XB` pads, as shown in the following figure. The values of the C_{X1} and C_{X2} capacitors depend on the selected crystal.

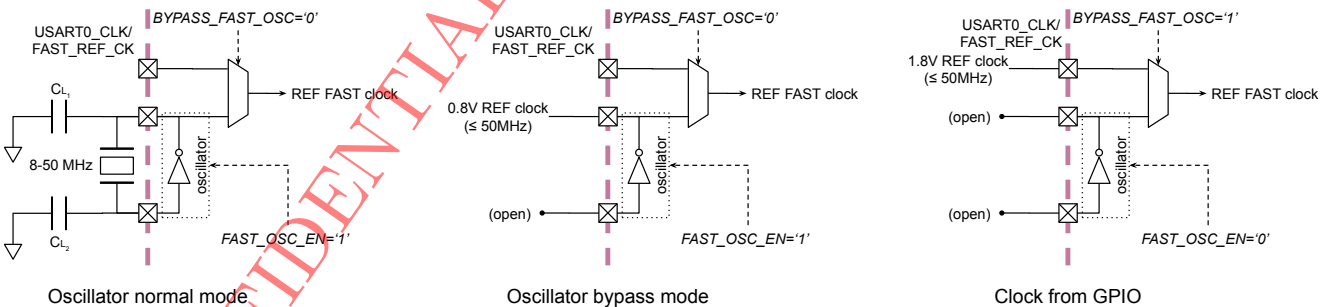


The slow oscillator can be selectively started or stopped through the SLOW_OSC_EN field of the REG_OSC_CTRL register, and may remain active during sleep modes, e.g. to use RTC as a wake-up source.

If the features that use the low-speed reference clock are not needed by the target application, then the external crystal and capacitors may be safely removed. In this case the `SLOW_XTAL_XA` should be tied to GND, and `SLOW_XTAL_XB` should be left floating.

4.2.2 High-Speed Reference Clock

GAP9 uses an internal fast reference clock up to 50MHz, which is the master clock for the generation of the different system clocks introduced below. This clock is referred to as the *REF FAST clock* in the following. The REF FAST clock is either generated by a dedicated embedded oscillator, or is directly provided through a chip pad (see the figure below).



When using the embedded oscillator, the REF FAST clock is either generated from an external crystal connected to the `FAST_XTAL_XA` and `FAST_XTAL_XB` pads (normal mode), or from a full-swing 0.8V external clock signal (bypass mode). When using a crystal, the values of the C_{L1} and C_{L2} capacitors shown in the figure depend on the selected crystal. No other external component is needed for clock generation.

When the oscillator is used in bypass mode, the external reference clock must be connected to `FAST_XTAL_XA`, and `FAST_XTAL_XB` must be left floating. The used oscillator does no require any specific configuration to operate with a given crystal or in bypass mode. Note that this bypass mode does not ensure a properly balanced duty cycle of the REF FAST clock; therefore specific care must be taken to select the settings of the dividers when using REF FAST clock to directly synchronize peripherals like USARTs and SAIs.

To save power, the REF FAST clock oscillator can be disabled through FAST_OSC_EN field of the REG_OSC_CTRL control register. In this case it is still possible to operate the chip, with limited frequency precision, by using the FLL in open-loop mode to generate system clocks (see the FLL section below).

If the FAST REF clock is to be provided through a chip pad, then a full-swing 1.8V signal must be sent through the `USART0_CLK/FAST_REF_CK` pad, which is then reserved for this use. This pad must be configured in GPIO input mode when an external clock is provided. Since this pad can be switched off, the input signal must only be provided when the I/O bank is switched on. This mode is selected by setting the BYPASS_FAST_OSC field of the REG_OSC_CTRL register to 1.

4.2.3 System Clocks

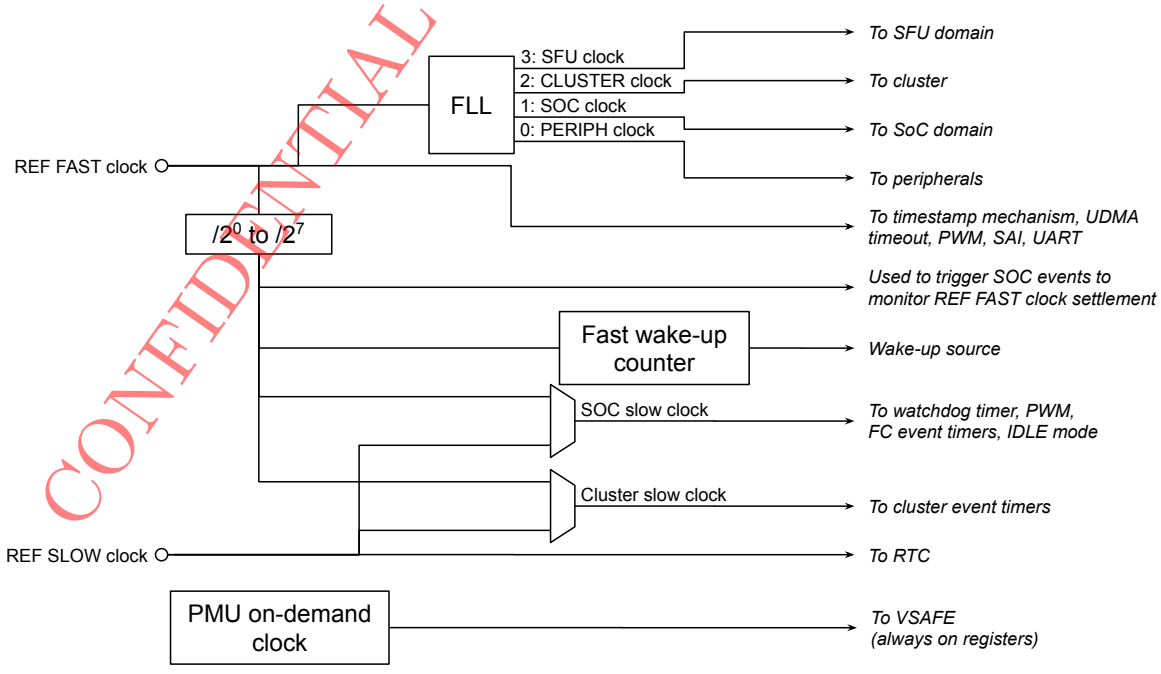
On top of the reference clocks decribed in the previous section, GAP9 uses 5 main system clocks (see the figure below):

- a divided version of the REF FAST clock, used as an intermediate speed reference clock,
- the SOC clock, which is the main clock for the SOC domain,
- the CLUSTER clock, which is used to clock the acceleration cluster,
- the SFU clock, which clocks the SFU,
- the PERIPH clock, which is used to generate the clocks of I/O interface IPs[1].

The divided version of REF FAST clock is generated by a clock divider which applies a configurable division by a power of 2 (see CLK_DIV_REF_FAST_POW2 register). Its frequency depends on the quartz or external oscillator used for REF CLOCK, and must be configured to remain below 12.5MHz. This clock is used as a time reference for the fast wake-up counter. It can also replace the REF SLOW clock for watchdog, IDLE and event timers, by proper configuration of the REF_CLK_MUX register. This is useful for example when the REF SLOW clock it is not available.

The SOC, CLUSTER, SFU and PERIPH clocks are generated by a *quadruple* FLL (frequency-locked loop), either in open-loop mode or based on REF FAST clock. When the REF FAST clock is to be used, the software can monitor its proper settlement before configuring the FLL, by using a dedicated event signal. The FLL is detailed in the next section.

Being always powered, the SAFE domain only supports low speed operation, for power consumption considerations. It only requires a clock when a write operation is made to its configuration registers, and benefits the on-demand clock mechanism of the PMU: when a write access is done, the PMU temporarily generates a few MHz clock which is used to store the configuration into the SAFE domain registers. The clock is then switched off.



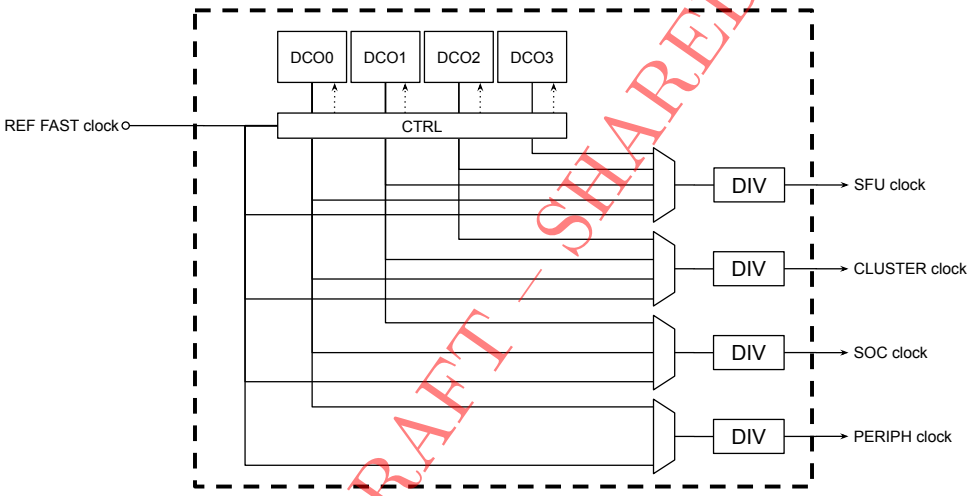
[1] SAI and UART interfaces may also use REF clock as a clock source for higher precision (improved jitter).

4.2.4 FLL

4.2.4.1 Overview

The next figure shows a simplified diagram of GAP9 FLL. The FLL embeds 4 independent digitally controlled oscillators (DCOs). When enabled, a DCO generates a clock, the frequency of which depends on a control parameter (input code). The FLL offers two different modes of operation:

- *Open-loop* is an unregulated mode, in which the software directly configures the DCO input code. In this mode the actual frequency depends on the operating conditions (process, voltage, temperature), therefore it cannot be used to generate precise clock frequencies. This mode is useful at startup, because it allows to start the system when the REF FAST clock is not yet available.
- *Closed-loop* mode uses a feedback loop impemented by the FLL controller in order to reach a target frequency configured by the software. In this mode, the FLL counts the number of DCO oscillations during a defined number of REF FAST clock periods (the integration period), and adjusts the DCO input code to reach target configuration. This mode is the one to be used for nominal operation, since it allows for precise clock settings.



FLL outputs are then selected from a DCO output or from the REF FAST clock, and a configurable integer division ratio is applied (DIV block). As shown in the figure, usable DCOs depend on the output: as an example the PERIPH clock can be generated from the first DCO only, whereas any DCO may be used for SFU clock. Those options are flexible enough to handle all possible use cases in terms of number of DCOs used:

- A single DCO is active and used for every output clock, possibly with different output dividers, or
- Each output uses a different DCO to generate completely independent clocks, or
- Any intermediate case with 2 or more clocks sharing a DCO.

Note: Write access to FLL configuration registers requires the REF FAST clock to be on.

4.2.4.2 Considerations on Open-Loop Mode

Open-loop mode is used at chip wake-up, in order to start FC operations without waiting for the REF FAST clock to be available. It can also be used on request from the application software for specific use cases.

A typical example is the use of an external clock source, which might not always be available. Since the REF FAST clock is required to configure FLL registers, it is mandatory to switch to open-loop mode before the external clock source is stopped, and to enable closed-loop again only after the REF FAST clock has stabilized.

Open-loop generated clock can be used to clock internal logic as well as interfaces to external components, still taking into account that the precision of the produced clock is limited and depends on operating conditions.

Output frequency – before application of the output stage divider – can be approximated by the following equation, with a ~25% (**TBC**) uncertainty:

$$f_{\text{DCO}} = \frac{43.734+2.734\times\text{DCO\_CODE}}{2} \text{ MHz}$$

4.2.4.3 Configuration in Closed-Loop Mode

In closed-loop mode, the configuration of a DCO output frequency is mainly done through the ITG_PER field of its F*CR1 register and the MFI field of its F*CR2 register:

$$f_{\text{DCO}} = f_{\text{REF FAST}} \times \frac{\text{MFI}}{\text{ITG\_PER}+1} \text{ MHz}$$

Other parameters in F*CR1 can be used to tune the behavior of the feedback loop (gain, refresh rate, target precision, etc.).

Note: The FLL design is based on the assumption that the frequency of a DCO is at least twice the frequency of the REF FAST clock: $f_{\text{DCO}} > 2 \times f_{\text{REF FAST}}$.

Note: For a good precision of the counting of DCO oscillations during the integration period, MFI should be in the 100's to 1000's range. ITG_PER must be high enough to enable meeting this requirement.

Chapter 5

Device Components Description

5.1 Cores

5.1.1 Standard RiscV ISA

5.1.1.1 RiscV I

For the sake of completeness we reproduce here part of the official Riscv documentation: https://content.riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
imm[31:12]				rd	0110111	lui rd,imm
imm[31:12]				rd	0010111	auipc rd,imm
imm[20|10:1|11|19:12]				rd	1101111	jal rd,imm
imm[11:0]		rs1	000	rd	0000011	jalr rd,imm
imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	beq rs1,rs2,imm
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	bne rs1,rs2,imm
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	blt rs1,rs2,imm
imm[12|10:5]	rs2	rs1	101	imm[4:1|11]	1100011	bge rs1,rs2,imm
imm[12|10:5]	rs2	rs1	110	imm[4:1|11]	1100011	bltu rs1,rs2,imm
imm[12|10:5]	rs2	rs1	111	imm[4:1|11]	1100011	bgeu rs1,rs2,imm
imm[11:0]		rs1	000	rd	0000011	lb rd,rs1,imm
imm[11:0]		rs1	001	rd	0000011	lh rd,rs1,imm
imm[11:0]		rs1	010	rd	0000011	lw rd,rs1,imm
imm[11:0]		rs1	100	rd	0000011	lbu rd,rs1,imm
imm[11:0]		rs1	101	rd	0000011	lhu rd,rs1,imm
Imm[11:5]	rs2	rs1	000	Imm[4:0]	0100011	sb rs1,rs2,imm
Imm[11:5]	rs2	rs1	001	Imm[4:0]	0100011	sh rs1,rs2,imm
Imm[11:5]	rs2	rs1	010	Imm[4:0]	0100011	sw rs1,rs2,imm
imm[11:0]		rs1	000	rd	0010011	addi rd,rs1,imm
imm[11:0]		rs1	010	rd	0010011	slti rd,rs1,imm
imm[11:0]		rs1	011	rd	0010011	sltiu rd,rs1,imm
imm[11:0]		rs1	100	rd	0010011	xori rd,rs1,imm
imm[11:0]		rs1	110	rd	0010011	ori rd,rs1,imm
imm[11:0]		rs1	111	rd	0010011	andi rd,rs1,imm
0000000	shamt	rs1	001	rd	0010011	slli rd,rs1,shamt
0000000	shamt	rs1	101	rd	0010011	srli rd,rs1,shamt
0100000	shamt	rs1	101	rd	0010011	srai rd,rs1,shamt

31:25	24:20	19:15	14:12	11:7	6:0	
0000000	rs2	rs1	000	rd	0110011	add rd,rs1,rs2
0100000	rs2	rs1	000	rd	0110011	sub rd,rs1,rs2
0000000	rs2	rs1	001	rd	0110011	sll rd,rs1,rs2
0000000	rs2	rs1	010	rd	0110011	slt rd,rs1,rs2
0000000	rs2	rs1	011	rd	0110011	sltu rd,rs1,rs2
0000000	rs2	rs1	100	rd	0110011	xor rd,rs1,rs2
0000000	rs2	rs1	101	rd	0110011	srlrd,rs1,rs2
0100000	rs2	rs1	101	rd	0110011	sra rd,rs1,rs2
0000000	rs2	rs1	110	rd	0110011	or rd,rs1,rs2
0000000	rs2	rs1	111	rd	0110011	and rd,rs1,rs2
000000000000		00000	000	00000	1110011	ecall
000000000001		00000	000	00000	1110011	ebreak

5.1.1.2 RiscV M

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
1	rs2	rs1	000	rd	0110011	mul rd,rs1,rs2
1	rs2	rs1	001	rd	0110011	mulh rd,rs1,rs2
1	rs2	rs1	010	rd	0110011	mulhsu rd,rs1,rs2
1	rs2	rs1	011	rd	0110011	mulhu rd,rs1,rs2
1	rs2	rs1	100	rd	0110011	div rd,rs1,rs2
1	rs2	rs1	101	rd	0110011	divu rd,rs1,rs2
1	rs2	rs1	110	rd	0110011	rem rd,rs1,rs2
1	rs2	rs1	111	rd	0110011	remu rd,rs1,rs2

5.1.1.3 RiscV C

See https://content.riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf

5.1.2 ISA Extension. Post modified load and store

5.1.2.1 Load Instructions

Mnemonic	Description
Register-Immediate Loads with Post-Increment	
p.lb rD, Imm(rs1!)	rD = Sext(Mem8(rs1)); rs1 += Imm[11:0]
p.lbu rD, Imm(rs1!)	rD = Zext(Mem8(rs1)); rs1 += Imm[11:0]
p.lh rD, Imm(rs1!)	rD = Sext(Mem16(rs1)); rs1 += Imm[11:0]
p.lhu rD, Imm(rs1!)	rD = Zext(Mem16(rs1)); rs1 += Imm[11:0]
p.lw rD, Imm(rs1!)	rD = Mem32(rs1); rs1 += Imm[11:0]
Register-Register Loads with Post-Increment	
p.lb rD, rs2(rs1!)	rD = Sext(Mem8(rs1)); rs1 += rs2
p.lbu rD, rs2(rs1!)	rD = Zext(Mem8(rs1)); rs1 += rs2
p.lh rD, rs2(rs1!)	rD = Sext(Mem16(rs1)); rs1 += rs2
p.lhu rD, rs2(rs1!)	rD = Zext(Mem16(rs1)); rs1 += rs2
p.lw rD, rs2(rs1!)	rD = Mem32(rs1); rs1 += rs2
Register-Register Loads	
p.lb rD, rs2(rs1)	rD = Sext(Mem8(rs1 + rs2))
p.lbu rD, rs2(rs1)	rD = Zext(Mem8(rs1 + rs2))
p.lh rD, rs2(rs1)	rD = Sext(Mem16(rs1 + rs2))
p.lhu rD, rs2(rs1)	rD = Zext(Mem16(rs1 + rs2))
p.lw rD, rs2(rs1)	rD = Mem32(rs1 + rs2)

5.1.2.2 Store instructions

Mnemonic	Description
Register-Immediate Stores with Post-Increment	
p.sb rs2, Imm(rs1!)	Mem8(rs1) = rs2; rs1 += Imm[11:0]
p.sh rs2, Imm(rs1!)	Mem16(rs1) = rs2; rs1 += Imm[11:0]
p.sw rs2, Imm(rs1!)	Mem32(rs1) = rs2; rs1 += Imm[11:0]
Register-Register Stores with Post-Increment	
p.sb rs2, rs3(rs1!)	Mem8(rs1) = rs2; rs1 += rs3
p.sh rs2, rs3(rs1!)	Mem16(rs1) = rs2; rs1 += rs3
p.sw rs2, rs3(rs1!)	Mem32(rs1) = rs2; rs1 += rs3
Register-Register Stores	
p.sb rs2, rs3(rs1)	Mem8(rs1 + rs3) = rs2
p.sh rs2 rs3(rs1)	Mem16(rs1 + rs3) = rs2
p.sw rs2, rs3(rs1)	Mem32(rs1 + rs3) = rs2

5.1.2.3 Encoding

31:20	19:15	14:12	11:7	6:0	
imm[11:0]	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
offset	base	000	dest	000 1011	p.lb rD, Imm(rs1!)
offset	base	100	dest	000 1011	p.lbu rD, Imm(rs1!)
offset	base	001	dest	000 1011	p.lh rD, Imm(rs1!)
offset	base	101	dest	000 1011	p.lhu rD, Imm(rs1!)
offset	base	010	dest	000 1011	p.lw rD, Imm(rs1!)

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
000 0000	offset	base	111	dest	000 1011	p.lb rD, rs2(rs1!)
010 0000	offset	base	111	dest	000 1011	p.lbu rD, rs2(rs1!)
000 1000	offset	base	111	dest	000 1011	p.lh rD, rs2(rs1!)
010 1000	offset	base	111	dest	000 1011	p.lhu rD, rs2(rs1!)
001 0000	offset	base	111	dest	000 1011	p.lw rD, rs2(rs1!)

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
000 0000	offset	base	111	dest	000 0011	p.lb rD, rs2(rs1)
010 0000	offset	base	111	dest	000 0011	p.lbu rD, rs2(rs1)
000 1000	offset	base	111	dest	000 0011	p.lh rD, rs2(rs1)
010 1000	offset	base	111	dest	000 0011	p.lhu rD, rs2(rs1)
001 0000	offset	base	111	dest	000 0011	p.lw rD, rs2(rs1)

31:25	24:20	19:15	14:12	11:7	6:0	
imm[11:5]	**rs2**	**rs1**	**funct3**	**imm[4:0]**	**opcode**	**instruction**
offset[11:5]	src	base	000	offset[4:0]	010 1011	p.sb rs2, Imm(rs1!)
offset[11:5]	src	base	001	offset[4:0]	010 1011	p.sh rs2, Imm(rs1!)
offset[11:5]	src	base	010	offset[4:0]	010 1011	p.sw rs2, Imm(rs1!)

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rs3**	**opcode**	**instruction**
000 0000	src	base	100	offset	010 1011	p.sb rs2, rs3(rs1!)
000 0000	src	base	101	offset	010 1011	p.sh rs2, rs3(rs1!)
000 0000	src	base	110	offset	010 1011	p.sw rs2, rs3(rs1!)

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rs3**	**opcode**	**instruction**
000 0000	src	base	100	offset	010 0011	p.sb rs2, rs3(rs1)
000 0000	src	base	101	offset	010 0011	p.sh rs2, rs3(rs1)
000 0000	src	base	110	offset	010 0011	p.sw rs2, rs3(rs1)

5.1.3 ISA Extension. Control flow instructions

5.1.3.1 Hardware loop instructions

Up to 2 levels of nested hardware loops are supported. The loop has to be setup before entering the loop body. For this purpose, there are two methods, either the long commands that separately set start- and end-addresses of the loop and the number of iterations, or the short command that does all of this in a single instruction. The short command has a limited range for the number of instructions contained in the loop and the loop must start in the next instruction after the setup instruction.

Loop number 0 has higher priority than loop number 1 in a nested loop configuration, meaning that loop 0 represents the inner loop.

A hardware loop is subject to the following constraints:

- Minimum of 2 instructions in the loop body.
- Loop counter has to be bigger than 0, since the loop body is always entered at least once.

Mnemonic		Description
Long Hardware Loop Setup instructions		
lp.starti	L, uimmL	lpstart[L] = PC + (uimmL << 1)
lp.endi	L, uimmL	lpend[L] = PC + (uimmL << 1)
lp.count	L, uimmL	lpcount[L] = rs1
lp.counti	L, uimmL	lpcount[L] = uimmL
Short Hardware Loop Setup instructions		
lp.setup	L, rs1, uimmL	lpstart[L] = pc+4; lpend[L] = pc+(uimmL << 1); lpcount[L] = rs1
lp.setupi	L, uimmL, uimmS	lpstart[L] = pc+4; lpend[L] = pc+(uimmS << 1); lpcount[L] = uimmL

5.1.3.2 Hardware loop instructions encoding

31:20	19:15	14:12	11:10	7	6:0	
uimmL[11:0]	**rs1**	**funct3**	**0**	**L**	**opcode**	**instruction**
uimmL[11:0]	0	0	0	L	111 1011	lp.starti L, uimmL
uimmL[11:0]	0	1	0	L	111 1011	lp.endi L, uimmL
0000 0000 0000	src1	10	0	L	111 1011	lp.count L, rs1
uimmL[11:0]	0	11	0	L	111 1011	lp.counti L, uimmL
uimmL[11:0]	src1	100	0	L	111 1011	lp.setup L, rs1, uimmL
uimmL[11:0]	uimmS[4:0]	101	0	L	111 1011	lp.setupi L, uimmS, uimmL

5.1.3.3 Branching instructions

Mnemonic	Description
p.beqimm rs1, Imm5, Imm12	Branch to PC + (Imm12 << 1) if rs1 is equal to Imm5. Imm5 is signed.
p.bneimm rs1, Imm5, Imm12	Branch to PC + (Imm12 << 1) if rs1 is not equal to Imm5. Imm5 is signed.

5.1.3.4 Branching instructions encoding

31:25	24:20	19:15	14:12	11:7	6:0	
imm12	**imm5**	**rs1**	**funct3**	**imm12**	**opcode**	**instruction**
[12][10:5]	[4:0]	src1	010	[4:1][11]	110 0011	p.beqimm rs1, Imm5, Imm12
[12][10:5]	[4:0]	src1	011	[4:1][11]	110 0011	p.bneimm rs1, Imm5, Imm12

5.1.4 ISA extension. 32b ALU instructions

5.1.4.1 32b Bit manipulation instructions

Mnemonic		Description
p.extract	rD, rs1, Is3, Is2	rD = Sext((rs1 & ((1 << Is3) - 1) << Is2) >> Is2) (*) Note: Is3 + Is2 must be <= 32
p.extractu	rD, rs1, Is3, Is2	rD = Zext((rs1 & ((1 << Is3) - 1) << Is2) >> Is2) (*) Note: Is3 + Is2 must be <= 32
p.extractr	rD, rs1, rs2	rD = Sext((rs1 & ((1 << rs2[9:5]) - 1) << rs2[4:0]) >> rs2[4:0]) (*) Note: rs2[9:5] + rs2[4:0] must be <= 32
p.extractur	rD, rs1, rs2	rD = Zext((rs1 & ((1 << rs2[9:5]) - 1) << rs2[4:0]) >> rs2[4:0]) (*) Note: rs2[9:5] + rs2[4:0] must be <= 32
p.insert	rD, rs1, Is3, Is2	rD = (rD & ~(rs1[Is3:0] << Is2)) | (rs1[Is3:0] << Is2) Note: Is3 + Is2 must be <= 32, the rest of the bits of rD are passed through and are not modified
p.insertr	rD, rs1, rs2	rD = (rD & ~(rs1[rs2[9:5]:0] << rs2[4:0])) | (rs1[rs2[9:5]:0] << rs2[4:0]) Note: rs2[9:5]+ rs2[4:0] must be <= 32, the rest of the bits of rD are passed through and are not modified
p.bclr	rD, rs1, Is3, Is2	rD = rs1 & ~(((1 << (Is3+1)) - 1) << Is2) Note: Is3 + Is2 must be <= 32
p.bclrr	rD, rs1, rs2	rD = rs1 & ~(((1 << (rs2[9:5]+1)) - 1) << rs2[4:0]) Note: rs2[9:5] + rs2[4:0] must be <= 32
p.bset	rD, rs1, Is3, Is2	rD = rs1 | (((1 << (Is3+1)) - 1) << Is2) Note: Is3 + Is2 must be <= 32
p.bsetr	rD, rs1, rs2	rD = rs1 | (((1 << (rs2[9:5]+1)) - 1) << rs2[4:0]) Note: rs2[9:5] + rs2[4:0] must be <= 32
p.ff1	rD, rs1	rD = bit position of the first bit set in rs1, starting from LSB. If bit 0 is set, rD will be 0. If only bit 31 is set, rD will be 31. If rs1 is 0, rD will be 32.
p.fl1	rD, rs1	rD = bit position of the last bit set in rs1, starting from MSB. If bit 31 is set, rD will be 31. If only bit 0 is set, rD will be 0. If rs1 is 0, rD will be 32.
p.clb	rD, rs1	rD = count leading bits of rs1. If rs1 is 0, rD will be 32. Note: This is the number of consecutive 1's or 0's from MSB.
p.cnt	rD, rs1	rD = population count of rs1, i.e. number of bits set in rs1
p.ror	rD, rs1, rs2	rD = RotateRight(rs1, rs2)
p.bitrev	rD, rs1, Is3, Is2	Given an input rs1. it returns a bit reversed representation assuming FFT on 2^Is2 points in Radix 2^Is3 Note: Is3 can be either 1, 2 or 3

(*) Sign extension is done over the extracted bit, i.e. the Is2-th bit.

5.1.4.2 32b Bit manipulation instructions encoding

31:30	29:25	24:20	19:15	14:12	11:7	6:0	
f2	**Is3[4:0]**	**Is2[4:0]**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
11	Luimm5[4:0]	Iuimm5[4:0]	src	000	dest	011 0011	p.extract rD, rs1, Is3, Is2
11	Luimm5[4:0]	Iuimm5[4:0]	src	001	dest	011 0011	p.extractu rD, rs1, Is3, Is2
11	Luimm5[4:0]	Iuimm5[4:0]	src	010	dest	011 0011	p.insert rD, rs1, Is3, Is2
11	Luimm5[4:0]	Iuimm5[4:0]	src	011	dest	011 0011	p.bclr rD, rs1, Is3, Is2
11	Luimm5[4:0]	Iuimm5[4:0]	src	100	dest	011 0011	p.bset rD, rs1, Is3, Is2
10	00000	src2	src1	000	dest	011 0011	p.extractr rD, rs1, rs2
10	00000	src2	src1	001	dest	011 0011	p.extractur rD, rs1, rs2
10	00000	src2	src1	010	dest	011 0011	p.insertr rD, rs1, rs2
10	00000	src2	src1	011	dest	011 0011	p.bclrr rD, rs1, rs2
10	00000	src2	scr1	100	dest	011 0011	p.bsetr rD, rs1, rs2
11	xxx,Luimm2[1:0]	Iuimm5[4:0]	src	101	dest	011 0011	p.bitrev rD, rs1, Is3, Is2

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
000 0100	src2	src1	101	dest	011 0011	p.ror rD, rs1, rs2
000 1000	00000	src1	000	dest	011 0011	p.ff1 rD, rs1
000 1000	00000	src1	001	dest	011 0011	p.fl1 rD, rs1
000 1000	00000	src1	010	dest	011 0011	p.clb rD, rs1
000 1000	00000	src1	011	dest	011 0011	p.cnt rD, rs1

5.1.4.3 32b General ALU instructions

5.1.4.4 32b General ALU instructions Encoding

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
000 0010	00000	src1	000	dest	011 0011	p.abs rD, rs1
000 0010	src2	src1	010	dest	011 0011	p.slet rD, rs1, rs2
000 0010	src2	src1	011	dest	011 0011	p.sletu rD, rs1, rs2
000 0010	src2	src1	100	dest	011 0011	p.min rD, rs1, rs2
000 0010	src2	src1	101	dest	011 0011	p.minu rD, rs1, rs2
000 0010	src2	src1	110	dest	011 0011	p.max rD, rs1, rs2
000 0010	src2	src1	111	dest	011 0011	p.maxu rD, rs1, rs2
000 1000	00000	src1	100	dest	011 0011	p.exths rD, rs1
000 1000	00000	src1	101	dest	011 0011	p.exthz rD, rs1
000 1000	00000	src1	110	dest	011 0011	p.extbs rD, rs1
000 1000	00000	src1	111	dest	011 0011	p.extbz rD, rs1

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**Is2[4:0]**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
000 1010	Iuimm5[4:0]	src1	001	dest	011 0011	p.clip rD, s1, Is2
000 1010	Iuimm5[4:0]	src1	010	dest	011 0011	p.clipu rD, s1, Is2
000 1010	src2	src1	010	dest	011 0011	p.clipr rD, s1, Is2
000 1010	src2	src1	110	dest	011 0011	p.clipur rD, s1, Is2

31:30	29:25	24:20	19:15	14:12	11:7	6:0	
f2	**Is3[4:0]**	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
00	Luimm5[4:0]	src2	src1	010	dest	101 1011	p.addN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	010	dest	101 1011	p.adduN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	110	dest	101 1011	p.addRN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	110	dest	101 1011	p.adduRN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	011	dest	101 1011	p.subN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	011	dest	101 1011	p.subuN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	111	dest	101 1011	p.subRN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	111	dest	101 1011	p.subuRN rD, rs1, rs2, Is3
01	00000	src2	src1	010	dest	101 1011	p.addNr rD, rs1, rs2
11	00000	src2	src1	010	dest	101 1011	p.adduNr rD, rs1, rs2
01	00000	src2	src1	110	dest	101 1011	p.addRNr rD, rs1, rs2
11	00000	src2	src1	110	dest	101 1011	p.adduRNr rD, rs1, rs2
01	00000	src2	src1	011	dest	101 1011	p.subNr rD, rs1, rs2
11	00000	src2	src1	011	dest	101 1011	p.subuN rD, rs1, rs2
01	00000	src2	src1	111	dest	101 1011	p.subRNr rD, rs1, rs2
11	00000	src2	src1	111	dest	101 1011	p.subuRNr rD, rs1, rs2

5.1.5 ISA extension. 32b Multiply and multiply and accumulate instructions

5.1.5.1 Instructions

Mnemonic		Description
32-Bit x 32-Bit Multiplication Operations		
p.mac	rD, rs1, rs2	rD = rD + rs1*rs2
p.msu	rD, rs1, rs2	rD = rD - rs1*rs2
16-Bit x 16-Bit Multiplication Operations		
p.muls	rD, rs1, rs2	rD[31:0] = Sext(rs1[15:0])*Sext(rs2[15:0])
p.mulhhs	rD, rs1, rs2	rD[31:0] = Sext(rs1[31:15])*Sext(rs2[31:15])
p.mulsN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[15:0])*Sext(rs2[15:0])) >>> Is3 Note: Arithmetic shift right
p.mulhhsN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[31:15])*Sext(rs2[31:15])) >>> Is3 Note: Arithmetic shift right
p.mulsRN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[15:0])*Sext(rs2[15:0]) + 2^(Is3-1)) >>> Is3 Note: Arithmetic shift right
p.mulhhsRN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[31:15])*Sext(rs2[31:15]) + 2^(Is3-1)) >>> Is3 Note: Arithmetic shift right
p.mulu	rD, rs1, rs2	rD[31:0] = Zext(rs1[15:0])*Zext(rs2[15:0])
p.mulhhu	rD, rs1, rs2	rD[31:0] = Zext(rs1[31:15])*Zext(rs2[31:15])
p.muluN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[15:0])*Zext(rs2[15:0])) >>> Is3 Note: Logical shift right
p.mulhhuN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[31:15])*Zext(rs2[31:15])) >>> Is3 Note: Logical shift right
p.muluRN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[15:0])*Zext(rs2[15:0]) + 2^(Is3-1)) >>> Is3 Note: Logical shift right
p.mulhhuRN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[31:15])*Zext(rs2[31:15]) + 2^(Is3-1)) >>> Is3 Note: Logical shift right
16-Bit x 16-Bit Multiply-Accumulate Operations		
p.macsN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[15:0])*Sext(rs2[15:0]) + rD) >>> Is3 Note: Arithmetic shift right
p.machhsN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[31:15])*Sext(rs2[31:15]) + rD) >>> Is3 Note: Arithmetic shift right
p.macsRN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[15:0])*Sext(rs2[15:0]) + rD + 2^(Is3-1)) >>> Is3 Note: Arithmetic shift right
p.machhsRN	rD, rs1, rs2, Is3	rD[31:0] = (Sext(rs1[31:15])*Sext(rs2[31:15]) + rD + 2^(Is3-1)) >>> Is3 Note: Arithmetic shift right
p.macuN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[15:0])*Zext(rs2[15:0]) + rD) >>> Is3 Note: Logical shift right
p.machhuN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[31:15])*Zext(rs2[31:15]) + rD) >>> Is3 Note: Logical shift right
p.macuRN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[15:0])*Zext(rs2[15:0]) + rD + 2^(Is3-1)) >>> Is3 Note: Logical shift right
p.machhuRN	rD, rs1, rs2, Is3	rD[31:0] = (Zext(rs1[31:15])*Zext(rs2[31:15]) + rD + 2^(Is3-1)) >>> Is3 Note: Logical shift right

5.1.5.2 Encoding

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
010 0001	src2	src1	000	dest	011 0011	p.mac rD, rs1, rs2
010 0001	src2	src1	001	dest	011 0011	p.msu rD, rs1, rs2

31:30	29:25	24:20	19:15	14:12	11:7	6:0	
f2	**Is3[4:0]**	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
10	00000	src2	src1	000	dest	101 1011	p.muls rD, rs1, rs2
11	00000	src2	src1	000	dest	101 1011	p.mulhhs rD, rs1, rs2
10	Luimm5[4:0]	src2	src1	000	dest	101 1011	p.mulsN rD, rs1, rs2, Is3
11	Luimm5[4:0]	src2	src1	000	dest	101 1011	p.mulhhsN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	100	dest	101 1011	p.mulsRN rD, rs1, rs2, Is3
11	Luimm5[4:0]	src2	src1	100	dest	101 1011	p.mulhhsRN rD, rs1, rs2, Is3
00	00000	src2	src1	000	dest	101 1011	p.mulu rD, rs1, rs2
01	00000	src2	src1	000	dest	101 1011	p.mulhhu rD, rs1, rs2
00	Luimm5[4:0]	src2	src1	000	dest	101 1011	p.muluN rD, rs1, rs2, Is3
01	Luimm5[4:0]	src2	src1	000	dest	101 1011	p.mulhhuN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	100	dest	101 1011	p.muluRN rD, rs1, rs2, Is3
01	Luimm5[4:0]	src2	src1	100	dest	101 1011	p.mulhhuRN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	001	dest	101 1011	p.macsN rD, rs1, rs2, Is3
11	Luimm5[4:0]	src2	src1	001	dest	101 1011	p.machhsN rD, rs1, rs2, Is3
10	Luimm5[4:0]	src2	src1	101	dest	101 1011	p.macsRN rD, rs1, rs2, Is3
11	Luimm5[4:0]	src2	src1	101	dest	101 1011	p.machhsRN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	001	dest	101 1011	p.macuN rD, rs1, rs2, Is3
01	Luimm5[4:0]	src2	src1	001	dest	101 1011	p.machhuN rD, rs1, rs2, Is3
00	Luimm5[4:0]	src2	src1	101	dest	101 1011	p.macuRN rD, rs1, rs2, Is3
01	Luimm5[4:0]	src2	src1	101	dest	101 1011	p.machhuRN rD, rs1, rs2, Is3

5.1.6 ISA extension. 32b Vectorial/SIMD instructions

Vectorial instructions perform operations in a SIMD-like manner on multiple sub-word elements at the same time. This is done by segmenting the data path into smaller parts when 8 or 16-bit operations should be performed. Vectorial instructions are available in two flavors:

- 8-Bit, to perform four operations on the 4 bytes inside a 32-bit word at the same time
- 16-Bit, to perform two operations on the 2 half-words inside a 32-bit word at the same time.

Additionally, there are three modes that influence the second operand:

1. Normal mode, vector-vector operation. Both operands, from rs1 and rs2, are treated as vectors of bytes or half-words.
2. Scalar replication mode (.sc), vector-scalar operation. Operand 1 is treated as a vector, while operand 2 is treated as a scalar and replicated two or four times to form a complete vector. The LSP is used for this purpose.
3. Immediate scalar replication mode (.sci), vector-scalar operation. Operand 1 is treated as vector, while operand 2 is treated as a scalar and comes from an immediate. The immediate is either sign- or zero-extended, depending on the operation. If not specified, the immediate is sign-extended.

5.1.6.1 32b Vectorial/SIMD ALU instructions

Mnemonic	Description
General ALU Instructions	
pv.add[.sc,.sci]{.h,.b}	rD[i] = (rs1[i] + op2[i]) & 0xFFFF
pv.add{.div2,.div4,.div8}	rD[i] = ((rs1[i] + op2[i]) & 0xFFFF) >> {1,2,3}
pv.sub[.sc,.sci]{.h,.b}	rD[i] = (rs1[i] - op2[i]) & 0xFFFF
pv.sub{.div2,.div4,.div8}	rD[i] = ((rs1[i] - rs2[i]) & 0xFFFF) >> {1,2,3}
pv.subrotmj{/,.div2,.div4,.div8}	rD[0] = ((rs1[1] - rs2[1]) & 0xFFFF) >> {0,1,2,3} rD[1] = ((rs2[0] - rs1[0]) & 0xFFFF) >> {0,1,2,3}
pv.avg[.sc,.sci]{.h,.b}	rD[i] = ((rs1[i] + op2[i]) & {0xFFFF, 0xFF}) >> 1 Note: Arithmetic right shift
pv.avgu[.sc,.sci]{.h,.b}	rD[i] = ((rs1[i] + op2[i]) & {0xFFFF, 0xFF}) >> 1
pv.min[.sc,.sci]{.h,.b}	rD[i] = rs1[i] < op2[i] ? rs1[i] : op2[i]
pv.minu[.sc,.sci]{.h,.b}	rD[i] = rs1[i] < op2[i] ? rs1[i] : op2[i] Note: Immediate is zero-extended, comparison is unsigned
pv.max[.sc,.sci]{.h,.b}	rD[i] = rs1[i] > op2[i] ? rs1[i] : op2[i]
pv.maxu[.sc,.sci]{.h,.b}	rD[i] = rs1[i] > op2[i] ? rs1[i] : op2[i] Note: Immediate is zero-extended, comparison is unsigned
pv.srl[.sc,.sci]{.h,.b}	rD[i] = rs1[i] >> op2[i] Note: Immediate is zero-extended, shift is logical
pv.sra[.sc,.sci]{.h,.b}	rD[i] = rs1[i] >>> op2[i] Note: Immediate is zero-extended, shift is arithmetic
pv.sll[.sc,.sci]{.h,.b}	rD[i] = rs1[i] << op2[i] Note: Immediate is zero-extended, shift is logical
pv.or[.sc,.sci]{.h,.b}	rD[i] = rs1[i] | op2[i]
pv.xor[.sc,.sci]{.h,.b}	rD[i] = rs1[i] ^ op2[i]
pv.and[.sc,.sci]{.h,.b}	rD[i] = rs1[i] & op2[i]
pv.abs{.h,.b}	rD[i] = rs1 < 0 ? -rs1 : rs1
pv.cplxconj	rD[0] = rs1[0] rD[1] = -rs1[1]
pv.extract.h	rD = Sext(rs1[((I+1)*16)-1 : I*16])
pv.extract.b	rD = Sext(rs1[((I+1)*8)-1 : I*8])
pv.extractu.h	rD = Zext(rs1[((I+1)*16)-1 : I*16])
pv.extractu.b	rD = Zext(rs1[((I+1)*8)-1 : I*8])
pv.insert.h	rD[((I+1)*16-1:I*16] = rs1[15:0] Note: The rest of the bits of rD are untouched and keep their previous value
pv.insert.b	rD[((I+1)*8-1:I*8] = rs1[7:0] Note: The rest of the bits of rD are untouched and keep their previous value
Dot-Product Instructions	
pv.dotup[.sc,.sci].h	rD = rs1[0]*op2[0] + rs1[1]*op2[1] Note: All operations are unsigned
pv.dotup[.sc,.sci].b	rD = rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: All operations are unsigned
pv.dotusp[.sc,.sci].h	rD = rs1[0]*op2[0] + rs1[1]*op2[1] Note: rs1 is treated as unsigned, while rs2 is treated as signed
pv.dotusp[.sc,.sci].b	rD = rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: rs1 is treated as unsigned, while rs2 is treated as signed
pv.dotsp[.sc,.sci].h	rD = rs1[0]*op2[0] + rs1[1]*op2[1] Note: All operations are signed
pv.dotsp[.sc,.sci].b	rD = rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: All operations are signed
pv.sdotup[.sc,.sci].h	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] Note: All operations are unsigned

Mnemonic	Description
pv.sdotup[.sc,.sci].b	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: All operations are unsigned
pv.sdotusp[.sc,.sci].h	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] Note: rs1 is treated as unsigned, while rs2 is treated as signed
pv.sdotusp[.sc,.sci].b	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: rs1 is treated as unsigned, while rs2 is treated as signed
pv.sdotsp[.sc,.sci].h	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] Note: All operations are signed
pv.sdotsp[.sc,.sci].b	rD = rD + rs1[0]*op2[0] + rs1[1]*op2[1] + rs1[2]*op2[2] + rs1[3]*op2[3] Note: All operations are signed
pv.cplxmul.r.{/,.div2,.div4,.div8}	rD[15:0] = (rs1[0]*rs2[0] - rs1[1]*rs2[1]) >> {15,16,17,18} rD[31:16] = rD[31:16]
pv.cplxmul.i.{/,.div2,.div4,.div8}	rD[15:0] = (rs1[0]*rs2[1] + rs1[1]*rs2[0]) >> {15,16,17,18} rD[31:16] = rD[31:16]
Shuffle and Pack Instrustions	
pv.shuffle.h	rD[31:16] = rs1[(rs2[16]*16 + 15):rs2[16]*16] rD[15:0] = rs1[(rs2[0]*16 + 15):rs2[0]*16]
pv.shuffle.sci.h	rD[31:16] = rs1[(I1*16 + 15):I1*16] rD[15:0] = rs1[(I0*16 + 15):I0*16] Note: I1 and I0 represent bits 1 and 0 of the immediate
pv.shuffle.b	rD[31:24] = rs1[(rs2[25:24]*8 + 7):rs2[25:24]*8] rD[23:16] = rs1[(rs2[17:16]*8 + 7):rs2[17:16]*8] rD[15:8] = rs1[(rs2[9:8]*8 + 7):rs2[9:8]*8] rD[7:0] = rs1[(rs2[1:0]*8 + 7):rs2[1:0]*8]
pv.shuffleI0.sci.b	rD[31:24] = rs1[7:0] rD[23:16] = rs1[((I5:I4)*8 + 7):(I5:I4)*8] rD[15:8] = rs1[((I3:I2)*8 + 7):(I3:I2)*8] rD[7:0] = rs1[((I1:I0)*8 + 7):(I1:I0)*8]
pv.shuffleI1.sci.b	rD[31:24] = rs1[15:8] rD[23:16] = rs1[((I5:I4)*8 + 7):(I5:I4)*8] rD[15:8] = rs1[((I3:I2)*8 + 7):(I3:I2)*8] rD[7:0] = rs1[((I1:I0)*8 + 7):(I1:I0)*8]
pv.shuffleI2.sci.b	rD[31:24] = rs1[23:16] rD[23:16] = rs1[((I5:I4)*8 + 7):(I5:I4)*8] rD[15:8] = rs1[((I3:I2)*8 + 7):(I3:I2)*8] rD[7:0] = rs1[((I1:I0)*8 + 7):(I1:I0)*8]
pv.shuffleI3.sci.b	rD[31:24] = rs1[31:24] rD[23:16] = rs1[((I5:I4)*8 + 7):(I5:I4)*8] rD[15:8] = rs1[((I3:I2)*8 + 7):(I3:I2)*8] rD[7:0] = rs1[((I1:I0)*8 + 7):(I1:I0)*8]
pv.shuffle2.h	rD[31:16] = ((rs2[17] == 1) ? rs1 : rD)[(rs2[16]*16 + 15):rs2[16]*16] rD[15:0] = ((rs2[17] == 1) ? rs1 : rD)[(rs2[0]*16 + 15):rs2[0]*16]
pv.shuffle2.b	rD[31:24] = ((rs2[26] == 1) ? rs1 : rD)[(rs2[25:24]*8 + 7):rs2[25:24]*8] rD[23:16] = ((rs2[18] == 1) ? rs1 : rD)[(rs2[17:16]*8 + 7):rs2[17:16]*8] rD[15:8] = ((rs2[10] == 1) ? rs1 : rD)[(rs2[9:8]*8 + 7):rs2[9:8]*8] rD[7:0] = ((rs2[2] == 1) ? rs1 : rD)[(rs2[1:0]*8 + 7):rs2[1:0]*8]
pv.pack	rD[31:16] = rs1[15:0] rD[15:0] = rs2[15:0]
pv.pack.h	rD[31:16] = rs1[31:16] rD[15:0] = rs2[31:16]
pv.packhi.b	rD[31:24] = rs1[7:0] rD[23:16] = rs2[7:0] Note: The rest of the bits of rD are untouched and keep their previous value

Mnemonic	Description
pv.packlo.b	rD[15:8] = rs1[7:0] rD[7:0] = rs2[7:0] Note: The rest of the bits of rD are untouched and keep their previous value

5.1.6.2 32b Vectorial/SIMD ALU instructions encodings

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
funct5	**F**		**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
0 0000	0	0	src2	src1	000	dest	101 0111	pv.add.h rD, rs1, rs2
0 0000	0	0	src2	src1	100	dest	101 0111	pv.add.sc.h rD, rs1, rs2
0 0000	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.add.sci.h rD, rs1, Imm6
0 0000	0	0	src2	src1	001	dest	101 0111	pv.add.b rD, rs1, rs2
0 0000	0	0	src2	src1	101	dest	101 0111	pv.add.sc.b rD, rs1, rs2
0 0000	0	Imm6[5:0]		src1	111	dest	101 0111	pv.add.sci.b rD, rs1, Imm6
0 1011	1	x	src2	src1	01x	dest	101 0111	pv.add.div2 rD, rs1, rs2
0 1011	1	x	src2	src1	10x	dest	101 0111	pv.add.div4 rD, rs1, rs2
0 1011	1	x	src2	src1	11x	dest	101 0111	pv.add.div8 rD, rs1, rs2
0 0001	0	0	src2	src1	000	dest	101 0111	pv.sub.h rD, rs1, rs2
0 0001	0	0	src2	src1	100	dest	101 0111	pv.sub.sc.h rD, rs1, rs2
0 0001	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sub.sci.h rD, rs1, Imm6
0 0001	0	0	src2	src1	001	dest	101 0111	pv.sub.b rD, rs1, rs2
0 0001	0	0	src2	src1	101	dest	101 0111	pv.sub.sc.b rD, rs1, rs2
0 0001	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sub.sci.b rD, rs1, Imm6
0 1100	1	x	src2	src1	01x	dest	101 0111	pv.sub.div2 rD, rs1, rs2
0 1100	1	x	src2	src1	10x	dest	101 0111	pv.sub.div4 rD, rs1, rs2
0 1100	1	x	src2	src1	11x	dest	101 0111	pv.sub.div8 rD, rs1, rs2
0 1101	1	x	src2	src1	00x	dest	101 0111	pv.subrotmj rD, rs1, rs2
0 1101	1	x	src2	src1	01x	dest	101 0111	pv.subrotmj.div2 rD, rs1, rs2
0 1101	1	x	src2	src1	10x	dest	101 0111	pv.subrotmj.div4 rD, rs1, rs2
0 1101	1	x	src2	src1	11x	dest	101 0111	pv.subrotmj.div8 rD, rs1, rs2
0 0010	0	0	src2	src1	000	dest	101 0111	pv.avg.h rD, rs1, rs2
0 0010	0	0	src2	src1	100	dest	101 0111	pv.avg.sc.h rD, rs1, rs2
0 0010	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.avg.sci.h rD, rs1, Imm6
0 0010	0	0	src2	src1	001	dest	101 0111	pv.avg.b rD, rs1, rs2
0 0010	0	0	src2	src1	101	dest	101 0111	pv.avg.sc.b rD, rs1, rs2
0 0010	0	Imm6[5:0]		src1	111	dest	101 0111	pv.avg.sci.b rD, rs1, Imm6
0 0011	0	0	src2	src1	000	dest	101 0111	pv.avgu.h rD, rs1, rs2
0 0011	0	0	src2	src1	100	dest	101 0111	pv.avgu.sc.h rD, rs1, rs2
0 0011	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.avgu.sci.h rD, rs1, Imm6
0 0011	0	0	src2	src1	001	dest	101 0111	pv.avgu.b rD, rs1, rs2
0 0011	0	0	src2	src1	101	dest	101 0111	pv.avgu.sc.b rD, rs1, rs2
0 0011	0	Imm6[5:0]		src1	111	dest	101 0111	pv.avgu.sci.b rD, rs1, Imm6
0 0100	0	0	src2	src1	000	dest	101 0111	pv.min.h rD, rs1, rs2
0 0100	0	0	src2	src1	100	dest	101 0111	pv.min.sc.h rD, rs1, rs2
0 0100	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.min.sci.h rD, rs1, Imm6
0 0100	0	0	src2	src1	001	dest	101 0111	pv.min.b rD, rs1, rs2
0 0100	0	0	src2	src1	101	dest	101 0111	pv.min.sc.b rD, rs1, rs2
0 0100	0	Imm6[5:0]		src1	111	dest	101 0111	pv.min.sci.b rD, rs1, Imm6
0 0101	0	0	src2	src1	000	dest	101 0111	pv.minu.h rD, rs1, rs2
0 0101	0	0	src2	src1	100	dest	101 0111	pv.minu.sc.h rD, rs1, rs2
0 0101	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.minu.sci.h rD, rs1, Imm6
0 0101	0	0	src2	src1	001	dest	101 0111	pv.minu.b rD, rs1, rs2

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
0 0101	0	0	src2	src1	101	dest	101 0111	pv.minu.sc.b rD, rs1, rs2
0 0101	0	Imm6[5:0]		src1	111	dest	101 0111	pv.minu.sci.b rD, rs1, Imm6
0 0110	0	0	src2	src1	000	dest	101 0111	pv.max.h rD, rs1, rs2
0 0110	0	0	src2	src1	100	dest	101 0111	pv.max.sc.h rD, rs1, rs2
0 0110	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.max.sci.h rD, rs1, Imm6
0 0110	0	0	src2	src1	001	dest	101 0111	pv.max.b rD, rs1, rs2
0 0110	0	0	src2	src1	101	dest	101 0111	pv.max.sc.b rD, rs1, rs2
0 0110	0	Imm6[5:0]		src1	111	dest	101 0111	pv.max.sci.b rD, rs1, Imm6
0 0111	0	0	src2	src1	000	dest	101 0111	pv.maxu.h rD, rs1, rs2
0 0111	0	0	src2	src1	100	dest	101 0111	pv.maxu.sc.h rD, rs1, rs2
0 0111	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.maxu.sci.h rD, rs1, Imm6
0 0111	0	0	src2	src1	001	dest	101 0111	pv.maxu.b rD, rs1, rs2
0 0111	0	0	src2	src1	101	dest	101 0111	pv.maxu.sc.b rD, rs1, rs2
0 0111	0	Imm6[5:0]		src1	111	dest	101 0111	pv.maxu.sci.b rD, rs1, Imm6
0 1000	0	0	src2	src1	000	dest	101 0111	pv.srl.h rD, rs1, rs2
0 1000	0	0	src2	src1	100	dest	101 0111	pv.srl.sc.h rD, rs1, rs2
0 1000	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.srl.sci.h rD, rs1, Imm6
0 1000	0	0	src2	src1	001	dest	101 0111	pv.srl.b rD, rs1, rs2
0 1000	0	0	src2	src1	101	dest	101 0111	pv.srl.sc.b rD, rs1, rs2
0 1000	0	Imm6[5:0]		src1	111	dest	101 0111	pv.srl.sci.b rD, rs1, Imm6
0 1001	0	0	src2	src1	000	dest	101 0111	pv.sra.h rD, rs1, rs2
0 1001	0	0	src2	src1	100	dest	101 0111	pv.sra.sc.h rD, rs1, rs2
0 1001	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sra.sci.h rD, rs1, Imm6
0 1001	0	0	src2	src1	001	dest	101 0111	pv.sra.b rD, rs1, rs2
0 1001	0	0	src2	src1	101	dest	101 0111	pv.sra.sc.b rD, rs1, rs2
0 1001	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sra.sci.b rD, rs1, Imm6
0 1010	0	0	src2	src1	000	dest	101 0111	pv.sll.h rD, rs1, rs2
0 1010	0	0	src2	src1	100	dest	101 0111	pv.sll.sc.h rD, rs1, rs2
0 1010	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sll.sci.h rD, rs1, Imm6
0 1010	0	0	src2	src1	001	dest	101 0111	pv.sll.b rD, rs1, rs2
0 1010	0	0	src2	src1	101	dest	101 0111	pv.sll.sc.b rD, rs1, rs2
0 1010	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sll.sci.b rD, rs1, Imm6
0 1011	0	0	src2	src1	000	dest	101 0111	pv.or.h rD, rs1, rs2
0 1011	0	0	src2	src1	100	dest	101 0111	pv.or.sc.h rD, rs1, rs2
0 1011	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.or.sci.h rD, rs1, Imm6
0 1011	0	0	src2	src1	001	dest	101 0111	pv.or.b rD, rs1, rs2
0 1011	0	0	src2	src1	101	dest	101 0111	pv.or.sc.b rD, rs1, rs2
0 1011	0	Imm6[5:0]		src1	111	dest	101 0111	pv.or.sci.b rD, rs1, Imm6
0 1100	0	0	src2	src1	000	dest	101 0111	pv.xor.h rD, rs1, rs2
0 1100	0	0	src2	src1	100	dest	101 0111	pv.xor.sc.h rD, rs1, rs2
0 1100	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.xor.sci.h rD, rs1, Imm6
0 1100	0	0	src2	src1	001	dest	101 0111	pv.xor.b rD, rs1, rs2
0 1100	0	0	src2	src1	101	dest	101 0111	pv.xor.sc.b rD, rs1, rs2
0 1100	0	Imm6[5:0]		src1	111	dest	101 0111	pv.xor.sci.b rD, rs1, Imm6
0 1101	0	0	src2	src1	000	dest	101 0111	pv.and.h rD, rs1, rs2
0 1101	0	0	src2	src1	100	dest	101 0111	pv.and.sc.h rD, rs1, rs2
0 1101	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.and.sci.h rD, rs1, Imm6
0 1101	0	0	src2	src1	001	dest	101 0111	pv.and.b rD, rs1, rs2
0 1101	0	0	src2	src1	101	dest	101 0111	pv.and.sc.b rD, rs1, rs2
0 1101	0	Imm6[5:0]		src1	111	dest	101 0111	pv.and.sci.b rD, rs1, Imm6
0 1110	0	0	xxxxx	src1	000	dest	101 0111	pv.abs.h rD, rs1
0 1110	0	0	xxxxx	src1	001	dest	101 0111	pv.abs.b rD, rs1
0 1011	1	x	xxxxx	src1	000	dest	101 0111	pv.cplxconj rD, rs1

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
0 1111	0	Imm6[5:0]		src1	110	dest	101 0111	pv.extract.h rD, Imm6
0 1111	0	Imm6[5:0]		src1	111	dest	101 0111	pv.extract.b rD, Imm6
1 0010	0	Imm6[5:0]		src1	110	dest	101 0111	pv.extractu.h rD, Imm6
1 0010	0	Imm6[5:0]		src1	111	dest	101 0111	pv.extractu.b rD, Imm6
1 0110	0	Imm6[5:0]		src1	110	dest	101 0111	pv.insert.h rD, Imm6
1 0110	0	Imm6[5:0]		src1	111	dest	101 0111	pv.insert.b rD, Imm6
1 0000	0	0	src2	src1	000	dest	101 0111	pv.dotup.h rD, rs1, rs2
1 0000	0	0	src2	src1	100	dest	101 0111	pv.dotup.sc.h rD, rs1, rs2
1 0000	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.dotup.sci.h rD, rs1, Imm6
1 0000	0	0	src2	src1	001	dest	101 0111	pv.dotup.b rD, rs1, rs2
1 0000	0	0	src2	src1	101	dest	101 0111	pv.dotup.sc.b rD, rs1, rs2
1 0000	0	Imm6[5:0]		src1	111	dest	101 0111	pv.dotup.sci.b rD, rs1, Imm6
1 0001	0	0	src2	src1	000	dest	101 0111	pv.dotusp.h rD, rs1, rs2
1 0001	0	0	src2	src1	100	dest	101 0111	pv.dotusp.sc.h rD, rs1, rs2
1 0001	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.dotusp.sci.h rD, rs1, Imm6
1 0001	0	0	src2	src1	001	dest	101 0111	pv.dotusp.b rD, rs1, rs2
1 0001	0	0	src2	src1	101	dest	101 0111	pv.dotusp.sc.b rD, rs1, rs2
1 0001	0	Imm6[5:0]		src1	111	dest	101 0111	pv.dotusp.sci.b rD, rs1, Imm6
1 0011	0	0	src2	src1	000	dest	101 0111	pv.dotsp.h rD, rs1, rs2
1 0011	0	0	src2	src1	100	dest	101 0111	pv.dotsp.sc.h rD, rs1, rs2
1 0011	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.dotsp.sci.h rD, rs1, Imm6
1 0011	0	0	src2	src1	001	dest	101 0111	pv.dotsp.b rD, rs1, rs2
1 0011	0	0	src2	src1	101	dest	101 0111	pv.dotsp.sc.b rD, rs1, rs2
1 0011	0	Imm6[5:0]		src1	111	dest	101 0111	pv.dotsp.sci.b rD, rs1, Imm6
1 0100	0	0	src2	src1	000	dest	101 0111	pv.sdotup.h rD, rs1, rs2
1 0100	0	0	src2	src1	100	dest	101 0111	pv.sdotup.sc.h rD, rs1, rs2
1 0100	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sdotup.sci.h rD, rs1, Imm6
1 0100	0	0	src2	src1	001	dest	101 0111	pv.sdotup.b rD, rs1, rs2
1 0100	0	0	src2	src1	101	dest	101 0111	pv.sdotup.sc.b rD, rs1, rs2
1 0100	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sdotup.sci.b rD, rs1, Imm6
1 0101	0	0	src2	src1	000	dest	101 0111	pv.sdotusp.h rD, rs1, rs2
1 0101	0	0	src2	src1	100	dest	101 0111	pv.sdotusp.sc.h rD, rs1, rs2
1 0101	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sdotusp.sci.h rD, rs1, Imm6
1 0101	0	0	src2	src1	001	dest	101 0111	pv.sdotusp.b rD, rs1, rs2
1 0101	0	0	src2	src1	101	dest	101 0111	pv.sdotusp.sc.b rD, rs1, rs2
1 0101	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sdotusp.sci.b rD, rs1, Imm6
1 0111	0	0	src2	src1	000	dest	101 0111	pv.sdotsp.h rD, rs1, rs2
1 0111	0	0	src2	src1	100	dest	101 0111	pv.sdotsp.sc.h rD, rs1, rs2
1 0111	0	Imm6[5:0]s		src1	110	dest	101 0111	pv.sdotsp.sci.h rD, rs1, Imm6
1 0111	0	0	src2	src1	001	dest	101 0111	pv.sdotsp.b rD, rs1, rs2
1 0111	0	0	src2	src1	101	dest	101 0111	pv.sdotsp.sc.b rD, rs1, rs2
1 0111	0	Imm6[5:0]		src1	111	dest	101 0111	pv.sdotsp.sci.b rD, rs1, Imm6
0 1010	1	1	src2	Src1	00x	dest	101 0111	pv.cplxmul.r rD, rs1, rs2
0 1010	1	1	src2	Src1	01x	dest	101 0111	pv.cplxmul.r.div2 rD, rs1, rs2
0 1010	1	1	src2	Src1	10x	dest	101 0111	pv.cplxmul.r.div4 rD, rs1, rs2
0 1010	1	1	src2	Src1	11x	dest	101 0111	pv.cplxmul.r.div8 rD, rs1, rs2
0 1010	1	0	src2	Src1	00x	dest	101 0111	pv.cplxmul.i rD, rs1, rs2

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
0 1010	1	0	src2	Src1	01x	dest	101 0111	pv.cplxmul.i.div2 rD, rs1, rs2
0 1010	1	0	src2	Src1	10x	dest	101 0111	pv.cplxmul.i.div4 rD, rs1, rs2
0 1010	1	0	src2	Src1	11x	dest	101 0111	pv.cplxmul.i.div8 rD, rs1, rs2
1 1000	0	0	src2	src1	000	dest	101 0111	pv.shuffle.h rD, rs1, rs2
1 1000	0	Imm6[5:0]		src1	110	dest	101 0111	pv.shuffle.sci.h rD, rs1, Imm6
1 1000	0	0	src2	src1	001	dest	101 0111	pv.shuffle.b rD, rs1, rs2
1 1000	0	Imm6[5:0]		src1	111	dest	101 0111	pv.shuffleI0.sci.b rD, rs1, Imm6
1 1101	0	Imm6[5:0]		src1	111	dest	101 0111	pv.shuffleI1.sci.b rD, rs1, Imm6
1 1110	0	Imm6[5:0]		src1	111	dest	101 0111	pv.shuffleI2.sci.b rD, rs1, Imm6
1 1111	0	Imm6[5:0]		src1	111	dest	101 0111	pv.shuffleI3.sci.b rD, rs1, Imm6
1 1001	0	0	src2	src1	000	dest	101 0111	pv.shuffle2.h rD, rs1, rs2
1 1001	0	0	src2	src1	001	dest	101 0111	pv.shuffle2.b rD, rs1, rs2
1 1010	0	0	src2	src1	000	dest	101 0111	pv.pack rD, rs1, rs2
1 1010	0	1	src2	src1	000	dest	101 0111	pv.pack.h rD, rs1, rs2
1 1011	0	0	src2	src1	001	dest	101 0111	pv.packhi.b rD, rs1, rs2
1 1100	0	0	src2	src1	001	dest	101 0111	pv.packlo.b rD, rs1, rs2

5.1.6.3 32b Vectorial/SIMD compare instructions

Vectorial comparisons are done on individual bytes (.b) or half-words (.h), depending on the chosen mode. If the comparison result is true, all bits in the corresponding byte/half-word are set to 1. If the comparison result is false, all bits are set to 0. The default mode (no .sc, .sci) compares the lowest byte/half-word of the first operand with the lowest byte/half-word of the second operand, and so on. If the mode is set to scalar replication (.sc), always the lowest byte/half-word of the second operand is used for comparisons, thus instead of a vector comparison a scalar comparison is performed. In the immediate scalar replication mode (.sci), the immediate given to the instruction is used for the comparison.

Mnemonic		Description
pv.cmpeq[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] == op2 ? ‘1 : ‘0
pv.cmpne[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] != op2 ? ‘1 : ‘0
pv.cmpgt[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] > op2 ? ‘1 : ‘0
pv.cmpge[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] >= op2 ? ‘1 : ‘0
pv.cmplt[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] < op2 ? ‘1 : ‘0
pv.cmple[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] <= op2 ? ‘1 : ‘0
pv.cmpgtu[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] > op2 ? ‘1 : ‘0 Note: Unsigned comparison
pv.cmpgeu[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] >= op2 ? ‘1 : ‘0 Note: Unsigned comparison
pv.cmpltu[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] < op2 ? ‘1 : ‘0 Note: Unsigned comparison
pv.cmpleu[.sc,.sci]{.h,.b}	rD, rs1, {rs2, Imm6}	rD[i] = rs1[i] <= op2 ? ‘1 : ‘0 Note: Unsigned comparison

5.1.6.4 6.4 32b Vectorial/SIMD compare instructions encodings

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
funct5	**F**		**rs2**	**rs1**	**funct3**	**rd**	**opcode**	**instruction**
0 0000	1	0	src2	src1	000	dest	101 0111	pv.cmpeq.h rD, rs1, rs2
0 0000	1	0	src2	src1	100	dest	101 0111	pv.cmpeq.sc.h rD, rs1, rs2
0 0000	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpeq.sci.h rD, rs1, Imm6
0 0000	1	0	src2	src1	001	dest	101 0111	pv.cm peq.b rD, rs1, rs2
0 0000	1	0	src2	src1	101	dest	101 0111	pv.cmeq.sc.b rD, rs1, rs2
0 0000	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpeq.sci.b rD, rs1, Imm6
0 0001	1	0	src2	src1	000	dest	101 0111	pv.cmpne.h rD, rs1, rs2
0 0001	1	0	src2	src1	100	dest	101 0111	pv.cmpne.sc.h rD, rs1, rs2
0 0001	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpne.sci.h rD, rs1, Imm6
0 0001	1	0	src2	src1	001	dest	101 0111	pv.cmpne.b rD, rs1, rs2
0 0001	1	0	src2	src1	101	dest	101 0111	pv.cmpne.sc.b rD, rs1, rs2
0 0001	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpne.sci.b rD, rs1, Imm6
0 0010	1	0	src2	src1	000	dest	101 0111	pv.cmpgt.h rD, rs1, rs2
0 0010	1	0	src2	src1	100	dest	101 0111	pv.cmpgt.sc.h rD, rs1, rs2
0 0010	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpgt.sci.h rD, rs1, Imm6
0 0010	1	0	src2	src1	001	dest	101 0111	pv.cmpgt.b rD, rs1, rs2
0 0010	1	0	src2	src1	101	dest	101 0111	pv.cmpgt.sc.b rD, rs1, rs2
0 0010	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpgt.sci.b rD, rs1, Imm6
0 0011	1	0	src2	src1	000	dest	101 0111	pv.cmpge.h rD, rs1, rs2
0 0011	1	0	src2	src1	100	dest	101 0111	pv.cmpge.sc.h rD, rs1, rs2
0 0011	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpge.sci.h rD, rs1, Imm6
0 0011	1	0	src2	src1	001	dest	101 0111	pv.cmpge.b rD, rs1, rs2
0 0011	1	0	src2	src1	101	dest	101 0111	pv.cmpge.sc.b rD, rs1, rs2
0 0011	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpge.sci.b rD, rs1, Imm6
0 0100	1	0	src2	src1	000	dest	101 0111	pv.cmplt.h rD, rs1, rs2
0 0100	1	0	src2	src1	100	dest	101 0111	pv.cmplt.sc.h rD, rs1, rs2
0 0100	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmplt.sci.h rD, rs1, Imm6
0 0100	1	0	src2	src1	001	dest	101 0111	pv.cmplt.b rD, rs1, rs2
0 0100	1	0	src2	src1	101	dest	101 0111	pv.cmplt.sc.b rD, rs1, rs2
0 0100	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmplt.sci.b rD, rs1, Imm6
0 0101	1	0	src2	src1	000	dest	101 0111	pv.cmple.h rD, rs1, rs2
0 0101	1	0	src2	src1	100	dest	101 0111	pv.cmple.sc.h rD, rs1, rs2
0 0101	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmple.sci.h rD, rs1, Imm6
0 0101	1	0	src2	src1	001	dest	101 0111	pv.cmple.b rD, rs1, rs2
0 0101	1	0	src2	src1	101	dest	101 0111	pv.cmple.sc.b rD, rs1, rs2
0 0101	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmple.sci.b rD, rs1, Imm6
0 0110	1	0	src2	src1	000	dest	101 0111	pv.cmpgtu.h rD, rs1, rs2
0 0110	1	0	src2	src1	100	dest	101 0111	pv.cmpgtu.sc.h rD, rs1, rs2
0 0110	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpgtu.sci.h rD, rs1, Imm6
0 0110	1	0	src2	src1	001	dest	101 0111	pv.cmpgtu.b rD, rs1, rs2
0 0110	1	0	src2	src1	101	dest	101 0111	pv.cmpgtu.sc.b rD, rs1, rs2
0 0110	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpgtu.sci.b rD, rs1, Imm6
0 0111	1	0	src2	src1	000	dest	101 0111	pv.cmpgeu.h rD, rs1, rs2
0 0111	1	0	src2	src1	100	dest	101 0111	pv.cmpgeu.sc.h rD, rs1, rs2
0 0111	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpgeu.sci.h rD, rs1, Imm6
0 0111	1	0	src2	src1	001	dest	101 0111	pv.cmpgeu.b rD, rs1, rs2
0 0111	1	0	src2	src1	101	dest	101 0111	pv.cmpgeu.sc.b rD, rs1, rs2

31:27	26	25	24:20	19:15	14:12	11:7	6:0	
0 0111	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpgeu.sci.b rD, rs1, Imm6
0 1000	1	0	src2	src1	000	dest	101 0111	pv.cmpltu.h rD, rs1, rs2
0 1000	1	0	src2	src1	100	dest	101 0111	pv.cmpltu.sc.h rD, rs1, rs2
0 1000	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpltu.sci.h rD, rs1, Imm6
0 1000	1	0	src2	src1	001	dest	101 0111	pv.cmpltu.b rD, rs1, rs2
0 1000	1	0	src2	src1	101	dest	101 0111	pv.cmpltu.sc.b rD, rs1, rs2
0 1000	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpltu.sci.b rD, rs1, Imm6
0 1001	1	0	src2	src1	000	dest	101 0111	pv.cmpleu.h rD, rs1, rs2
0 1001	1	0	src2	src1	100	dest	101 0111	pv.cmpleu.sc.h rD, rs1, rs2
0 1001	1	Imm6[5:0]		src1	110	dest	101 0111	pv.cmpleu.sci.h rD, rs1, Imm6
0 1001	1	0	src2	src1	001	dest	101 0111	pv.cmpleu.b rD, rs1, rs2
0 1001	1	0	src2	src1	101	dest	101 0111	pv.cmpleu.sc.b rD, rs1, rs2
0 1001	1	Imm6[5:0]		src1	111	dest	101 0111	pv.cmpleu.sci.b rD, rs1, Imm6

Note: Imm6[5:0] is encoded as { Imm6[0], Imm6[5:1] }, LSB at the 25th bit of the instruction

5.1.7 ISA extension. 64b instructions

64b operations are supported using register pairing, e.g a 64b register is built from 2 adjacent 32b registers Rdi = (Ri,Ri+1)

Notation

1. rD, rS1, rS2: 32b registers, destination, first operand, second operand.
2. rDd, rS1d, rS2d: 64b registers, destination, first operand, second operand.
3. 64b registers are numbered from 0 to 30 (xd0 to xd30), xdi is the pair xi,xi+1. Register pair overlap is of course managed by the compiler.
4. Some instructions are consuming 64b inputs and producing 32b output.
5. Similarly some instructions are consuming 32b inputs and producing 64b output.
6. .d suffix appended to an instruction designates a 64b instruction.

5.1.7.1 64b general ALU instructions

Mnemonic		Description
add.d	rDd, rS1d, rS2d	rDd = (rS1d + rS2d)
sub.d	rDd, rS1d, rS2d	rDd = (rS1d - rS2d)
sll.d	rDd, rS1d, rS2d	rDd = (rS1d << rS2d)
slt.d	rD, rS1d, rS2d	rD = (rS1d < rS2d) ? 1 : 0 Note: Comparison is signed
sltu.d	rD, rS1d, rS2d	rD = (rS1d <u rS2d) ? 1 : 0 Note: Comparison is unsigned
xor.d	rDd, rS1d, rS2d	rDd = (rS1d ^ rS2d)
srl.d	rDd, rS1d, rS2d	rDd = (rS1d >>u rS2d) Note: Shift is unsigned
sra.d	rDd, rS1d, rS2d	rDd = (rS1d >> rS2d) Note: Shift is signed
or.d	rDd, rS1d, rS2d	rDd = (rS1d | rS2d)
and.d	rDd, rS1d, rS2d	rDd = (rS1d & rS2d)
Immediate Instructions		
slli.d srli.d	rDd, rS1d, shamt rDd, rS1d, shamt	rDd = (rS1d << shamt) rDd = (rS1d >>u shamt) Note: unsigned shift
srai.d	rDd, rS1d, shamt	rDd = (rS1d >> shamt) Note: signed shift
addi.d	rDd, rS1d, shamt	rDd = (rS1d + shamt) Note: shamt signed
slti.d	rD, rS1d, shamt	rD = (rS1d < shamt) ? 1 : 0 Note: Comparison is signed
sltiu.d	rD, rS1d, shamt	rD = (rS1d <u shamt) ? 1 : 0 Note: Comparison is unsigned
xori.d	rDd, rS1d, shamt	rDd = (rS1d ^ shamt)
ori.d	rDd, rS1d, shamt	rDd = (rS1d | shamt)
andi.d	rDd, rS1d, shamt	rDd = (rS1d & shamt)

5.1.7.2 64b general ALU instructions encoding

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
00 1 0000	rS2d	rS1d	000	rDd	01 100 11	add.d rDd, rS1d, rS2d
01 1 0000	rS2d	rS1d	000	rDd	01 100 11	sub.d rDd, rS1d, rS2d
00 1 0000	rS2d	rS1d	001	rDd	01 100 11	sll.d rDd, rS1d, rS2d
00 1 1000	rS2d	rS1d	010	rD	01 100 11	slt.d rD, rS1d, rS2d
00 1 1000	rS2d	rS1d	011	rD	01 100 11	sltu.d rD, rS1d, rS2d
00 1 0000	rS2d	rS1d	100	rDd	01 100 11	xor.d rDd, rS1d, rS2d
00 1 0000	rS2d	rS1d	101	rDd	01 100 11	srl.d rDd, rS1d, rS2d
01 1 0000	rS2d	rS1d	101	rDd	01 100 11	sra.d rDd, rS1d, rS2d
01 1 0000	rS2d	rS1d	110	rDd	01 100 11	or.d rDd, rS1d, rS2d
01 1 0000	rS2d	rS1d	111	rDd	01 100 11	and.d rDd, rS1d, rS2d

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**imm5**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
00 1 0000	shamt	rS1d	001	rDd	00 100 11	slli.d rDd, rS1d, shamt
00 1 0010	shamt	rS1d	101	rDd	00 100 11	srli.d rDd, rS1d, shamt
01 1 0010	shamt	rS1d	101	rDd	00 100 11	srai.d rDd, rS1d, shamt
00 1 0001	shamt	rS1d	001	rDd	00 100 11	addi.d rDd, rS1d, shamt
00 1 1000	shamt	rS1d	010	rD	00 110 11	slti.d rD, rS1d, shamt
00 1 1000	shamt	rS1d	011	rD	00 110 11	sltiu.d rD, rS1d, shamt
00 1 0000	shamt	rS1d	100	rDd	00 110 11	xori.d rDd, rS1d, shamt
00 1 0000	shamt	rS1d	110	rDd	00 110 11	ori.d rDd, rS1d, shamt
00 1 0000	shamt	rS1d	111	rDd	00 110 11	andi.d rDd, rS1d, shamt

5.1.7.3 64b extended ALU instructions

Mnemonic		**Description**
p.abs.d	rDd, rS1d	rDd = rS1d < 0 ? –rS1d : rS1d
p.seq.d	rD, rS1d, rS2d	rD = rS1d == rS2d ? 1 : 0
p.sne.d	rD, rS1d, rS2d	rD = rS1d != rS2d ? 1 : 0
p.slet.d	rD, rS1d, rS2d	rD = rS1d <= rS2d ? 1 : 0 Note: Comparison is signed
p.sletu.d	rD, rS1d, rS2d	rD = rS1d <= rS2d ? 1 : 0 Note: Comparison is unsigned
p.min.d	rDd, rS1d, rS2d	rDd = rS1d < rS2d ? rS1d : rS2d Note: Comparison is signed
p.minu.d	rDd, rS1d, rS2d	rDd = rS1d < rS2d ? rS1d : rS2d Note: Comparison is unsigned
p.max.d	rDd, rS1d, rS2d	rDd = rS1d < rS2d ? rS2d : rS1d Note: Comparison is signed
p.maxu.dxu.d	rDd, rS1d, rS2d	rDd = rS1d < rS2d ? rS2d : rS1d Note: Comparison is unsigned
p.cnt.d	rD, rS1d	rD = count bit1 (rS1d)
p.exths.d	rDd, rS1	rDd = Sext(rS1[15:0])
p.exthz.d	rDd, rS1	rDd = Zext(rS1[15:0])
p.extbs.d	rDd, rS1	rDd = Sext(rS1[7:0])
p.extbz.d	rDd, rS1	rDd = Zext(rS1[7:0])
p.extws.d	rDd, rS1	rDd = Sext(rS1[31:0])
p.extwz.d	rDd, rS1	rDd = Zext(rS1[31:0])

5.1.7.4 64b extended ALU instructions encoding

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
00 1 0010	00000	rS1d	000	rDd	01 100 11	p.abs.d rDd, rS1d
00 1 1011	rS2d	rS1d	010	rD	01 100 11	p.seq.d rDd, rS1d, rS2d
00 1 1010	rS2d	rS1d	010	rD	01 100 11	p.slet.d rD, rS1d, rS2d
00 1 1010	rS2d	rS1d	011	rD	01 100 11	p.sletu.d rD, rS1d, rS2d
00 1 1011	rS2d	rS1d	011	rD	01 100 11	p.sne.d rD, rS1d, rS2d
00 1 0010	rS2d	rS1d	100	rDd	01 100 11	p.min.d rDd, rS1d, rS2d
00 1 0010	rS2d	rS1d	101	rDd	01 100 11	p.minu.d rDd, rS1d, rS2d
00 1 0010	rS2d	rS1d	110	rDd	01 100 11	p.max.d rDd, rS1d, rS2d
00 1 0010	rS2d	rS1d	111	rDd	01 100 11	p.maxu.d rDd, rS1d, rS2d
00 1 1010	00000	rS1d	001	rD	01 100 11	p.cnt.d rD, rS1d
01 1 0010	00000	rS1	000	rDd	01 100 11	p.exths.d rDd, rS1
01 1 0010	00000	rS1	001	rDd	01 100 11	p.exthz.d rDd, rS1
01 1 0010	00000	rS1	010	rDd	01 100 11	p.extbs.d rDd, rS1
00 1 0010	00000	rS1	011	rDd	01 100 11	p.extbz.d rDd, rS1
01 1 0010	00000	rS1	100	rDd	01 100 11	p.extws.d rDd, rS1
01 1 0010	00000	rS1	101	rDd	01 100 11	p.extwz.d rDd, rS1

5.1.7.5 64b Multiply and Multiply and Accumulate instructions

Mnemonic		**Description**
p.mac.d	rDd, rS1, rS2	rDd = rDd + Sext(rS1) * Sext(rS2)
p.msu.d	rDd, rS1, rS2	rDd = rDd - Sext(rS1) * Sext(rS2)
p.macu.d	rDd, rS1, rS2	rDd = rDd + Zext(rS1) * Zext(rS2)
p.msuu.d	rDd, rS1, rS2	rDd = rDd - Zext(rS1) * Zext(rS2)
p.muls.d	rDd, rS1, rS2	rDd = Sext(rS1) * Sext(rS2)
p.mulu.d	rDd, rS1, rS2	rDd = Zext(rS1) * Zext(rS2)

5.1.7.6 64b Multiply and Multiply and Accumulate instructions encoding

31:25	24:20	19:15	14:12	11:7	6:0	
funct7	**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
01 1 1001	rS2	rS1	000	rDd	01 100 11	p.mac.d rDd, rS1, rS2
01 1 1001	rS2	rS1	001	rDd	01 100 11	p.msu.d rDd, rS1, rS2
01 1 1001	rS2	rS1	010	rDd	01 100 11	p.macu.d rDd, rS1, rS2
01 1 1001	rS2	rS1	011	rDd	01 100 11	p.msuu.d rDd, rS1, rS2
01 1 1001	rS2	rS1	100	rDd	01 100 11	p.muls.d rDd, rS1, rS2
01 1 1001	rS2	rS1	101	rDd	01 100 11	p.mulu.d rDd, rS1, rS2

5.1.8 Floating point instructions

IEEE 32b float ISA is directly derived from RiscV F ISA but no second dedicated register file is assumed. Instead the 32b integer register file is used. As a consequence load, store and move from/to float register file have been removed.

5.1.8.1 Floating point rounding modes

rm	Meaning
000	Round to Nearest, ties to Even
001	Round towards Zero
010	Round Down (towards $-\infty$)
011	Round Up (towards $+\infty$)
100	Round to Nearest, ties to Max Magnitude
101	Invalid, used for ALTF (non IEEE float16)
110	Invalid
111	In instruction's rm field, selects dynamic rounding mode

5.1.8.2 8.2 32b IEEE floating point instructions

For details about standard RiscV floating point instructions: https://content.riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf

31:27	26:25	24:20	19:15	14:12	11:7	6:0	
funct7		**rs2**	**rs1**	**funct3**	**rD**	**opcode**	**instruction**
rs3	**f2**						
rs3	00	rs2	rs1	rm	rd	1000011	fmadd.s rd,rs1,rs2,rs3
rs3	00	rs2	rs1	rm	rd	1000111	fmsub.s rd,rs1,rs2,rs3
rs3	00	rs2	rs1	rm	rd	1001011	fnmsub.s rd,rs1,rs2,rs3
rs3	00	rs2	rs1	rm	rd	1001111	fnmadd.s rd,rs1,rs2,rs3
0000000		rs2	rs1	rm	rd	1010011	fadd.s rd,rs1,rs2
0000100		rs2	rs1	rm	rd	1010011	fsub.s rd,rs1,rs2
0001000		rs2	rs1	rm	rd	1010011	fmul.s rd,rs1,rs2
0001100		rs2	rs1	rm	rd	1010011	fdiv.s rd,rs1,rs2
0101100		00000	rs1	rm	rd	1010011	fsqrt.s rd,rs1
0010000		rs2	rs1	000	rd	1010011	fsgnj.s rd,rs1,rs2
0010000		rs2	rs1	001	rd	1010011	fsgnjn.s rd,rs1,rs2
0010000		rs2	rs1	010	rd	1010011	fsgnjx.s rd,rs1,rs2
0010100		rs2	rs1	000	rd	1010011	fmin.s rd,rs1,rs2
0010100		rs2	rs1	001	rd	1010011	fmax.s rd,rs1,rs2
1100000		00000	rs1	rm	rd	1010011	fcvt.w.s rd,rs1
1100000		00001	rs1	rm	rd	1010011	fcvt.wu.s rd,rs1
1010000		rs2	rs1	010	rd	1010011	feq.s rd,rs1,rs2
1010000		rs2	rs1	001	rd	1010011	flt.s rd,rs1,rs2
1010000		rs2	rs1	000	rd	1010011	fle.s rd,rs1,rs2
1110000		00000	rs1	001	rd	1010011	fclass.s rd,rs1
1101000		00000	rs1	rm	rd	1010011	fcvt.s.w rd,rs1
1101000		00001	rs1	rm	rd	1010011	fcvt.s.wu rd,rs1

5.1.8.3 8.3 ISA extension. 16b IEEE floating point instructions, scalar

- The 16-bit half-precision floating-point representation is specified as binary16 in IEEE 754-2008. It is defined having 1 sign bit, 5 exponent bits and 10 mantissa bits. The exponent is biased by $2^{\text{expbits-1}}$ - 1 = 15, the mantissa is normalized and does not store the implicit bit. The canonical NaN as specified in the RiscV F standard extension for this format is the bit pattern 0x7e00.
- The floating-point control and status register fcsr is used as defined in the RiscV F standard extension.
- Rounding mode (rm) definition is the same as 32b IEEE.
- Format (fmt) is defined as 00 for 32b float and 10 for 16b float.

31:27	26:25	24:20	19:15	14:12	11:7	6:0	
funct5	**fmt**	**rs2**	**rs1**	**rm**	**rd**	**opcode**	**instruction**
rs3							
rs3	10	rs2	rs1	rm(*)	rd	1000011	fmadd.h rd,rs1,rs2,rs3
rs3	10	rs2	rs1	rm(*)	rd	1000111	fmsub.h rd,rs1,rs2,rs3
rs3	10	rs2	rs1	rm(*)	rd	1001011	fnmsub.h rd,rs1,rs2,rs3
rs3	10	rs2	rs1	rm(*)	rd	1001111	fnmadd.h rd,rs1,rs2,rs3
00000	10	rs2	rs1	rm(*)	rd	1010011	fadd.h rd,rs1,rs2
00001	10	rs2	rs1	rm(*)	rd	1010011	fsub.h rd,rs1,rs2
00010	10	rs2	rs1	rm(*)	rd	1010011	fmul.h rd,rs1,rs2
00011	10	rs2	rs1	rm(*)	rd	1010011	fdiv.h rd,rs1,rs2
01011	10	00000	rs1	rm(*)	rd	1010011	fsqrt.h rd,rs1
00100	10	rs2	rs1	000	rd	1010011	fsgnj.h rd,rs1,rs2
00100	10	rs2	rs1	001	rd	1010011	fsgnjn.h rd,rs1,rs2
00100	10	rs2	rs1	010	rd	1010011	fsgnjx.h rd,rs1,rs2
00101	10	rs2	rs1	000	rd	1010011	fmin.h rd,rs1,rs2
00101	10	rs2	rs1	001	rd	1010011	fmax.h rd,rs1,rs2
10100	10	rs2	rs1	010	rd	1010011	feq.h rd,rs1,rs2
10100	10	rs2	rs1	001	rd	1010011	flt.h rd,rs1,rs2
10100	10	rs2	rs1	000	rd	1010011	fle.h rd,rs1,rs2
11100	10	00000	rs1	001	rd	1010011	fclass.h rd,rs1
11000	10	00000	rs1	rm(*)	rd	1010011	fcvt.w.h rd,rs1
11000	10	00001	rs1	rm(*)	rd	1010011	fcvt.wu.h rd,rs1
11010	10	00000	rs1	rm(*)	rd	1010011	fcvt.h.w rd,rs1
11010	10	00001	rs1	rm(*)	rd	1010011	fcvt.h.wu rd,rs1
01000	00	00010	rs1	000	rd	1010011	fcvt.s.h rd,rs1
01000	10	00000	rs1	rm(*)	rd	1010011	fcvt.h.s rd,rs1

(*) Only valid rounding modes allowed (000-100, 111)

5.1.8.4 ISA extension. 16b Non IEEE floating point instructions, scalar

- The 16-bit alternative half-precision floating-point representation is specified as binary16alt in this specification. It is defined having 1 sign bit, 8 exponent bits and 7 mantissa bits. The exponent is biased by $2^{\text{expbits-1}}$ - 1 = 127, the mantissa is normalized and does not store the implicit bit. The canonical NaN as specified in the F standard extension for this format is the bit pattern 0x7fc0.
- The floating-point control and status register fcsr is used as defined in the F standard extension.
- Instructions that are affected by rounding modes works as follows:
  - When the format field fmt is set to [A]H (10)
    - If the rounding mode field rm holds one of the valid patterns as per the RiscV F standard extension (000-100 or 111), half-precision (binary16) is selected.
    - If the rounding mode is set to ALTF (101), operations will interpret the operands as alternative half-precision (binary16alt) values. In this case, rounding mode is applied from the frm field in fcsr.
  - Otherwise, ALTF (101) is treated as an invalid rounding mode as specified in the RiscV F standard extension.

- Instructions that are not affected by rounding modes but use the rm or rs2 field as part of their operation encoding (e.g. FMIN) are encoded as follows:
  - When the format field fmt is set to [A]H (10)
    - Operations on binary16 use the encoding in rm/rs2 as defined in the Xf16 extension.
    - Operations on binary16alt use a new encoding in rm/rs2 . This new encoding does not collide with other instructions.
  - Otherwise, the instructions with the new encoding in rm/rs2 are not defined/invalid.
- Rounding mode (rm) definition is the same as 32b IEEE
- Format (fmt) is defined as 00 for 32b float and 10 for 16b float

31:27	26:25	24:20	19:15	14:12	11:7	6:0	
funct5	**fmt**	**rs2**	**rs1**	**rm**	**rd**	**opcode**	**instruction**
rs3							
rs3	10	rs2	rs1	101	rd	1000011	fmadd.ah rd,rs1,rs2,rs3
rs3	10	rs2	rs1	101	rd	1000111	fmsub.ah rd,rs1,rs2,rs3
rs3	10	rs2	rs1	101	rd	1001011	fnmsub.ah rd,rs1,rs2,rs3
rs3	10	rs2	rs1	101	rd	1001111	fnmadd.ah rd,rs1,rs2,rs3
00000	10	rs2	rs1	101	rd	1010011	fadd.ah rd,rs1,rs2
00001	10	rs2	rs1	101	rd	1010011	fsub.ah rd,rs1,rs2
00010	10	rs2	rs1	101	rd	1010011	fmul.ah rd,rs1,rs2
00011	10	rs2	rs1	101	rd	1010011	fdiv.ah rd,rs1,rs2
01011	10	000	rs1	101	rd	1010011	fsqrt.ah rd,rs1
00100	10	rs2	rs1	100	rd	1010011	fsgnj.ah rd,rs1,rs2
00100	10	rs2	rs1	101	rd	1010011	fsgnjn.ah rd,rs1,rs2
00100	10	rs2	rs1	110	rd	1010011	fsgnjx.ah rd,rs1,rs2
00101	10	rs2	rs1	100	rd	1010011	fmin.ah rd,rs1,rs2
00101	10	rs2	rs1	101	rd	1010011	fmax.ah rd,rs1,rs2
10100	10	rs2	rs1	110	rd	1010011	feq.ah rd,rs1,rs2
10100	10	rs2	rs1	101	rd	1010011	flt.ah rd,rs1,rs2
10100	10	rs2	rs1	100	rd	1010011	fle.ah rd,rs1,rs2
11100	10	000	rs1	101	rd	1010011	fclass.ah rd,rs1
11000	10	000	rs1	101	rd	1010011	fcvt.w.ah rd,rs1
11000	10	001	rs1	101	rd	1010011	fcvt.wu.ah rd,rs1
11010	10	000	rs1	101	rd	1010011	fcvt.ah.w rd,rs1
11010	10	001	rs1	101	rd	1010011	fcvt.ah.wu rd,rs1
01000	00	110	rs1	000	rd	1010011	fcvt.s.ah rd,rs1
01000	10	000	rs1	101	rd	1010011	fcvt.ah.s rd,rs1
01000	10	110	rs1	rm(*)	rd	1010011	fcvt.h.ah rd,rs1
01000	10	010	rs1	101	rd	1010011	fcvt.ah.h rd,rs1

(*) Only valid rounding modes allowed (000-100, 111)

5.1.8.5 ISA extension, 16b IEEE floating point instructions, vector

- The 16-bit half-precision floating-point representation is specified as binary16 in IEEE 754-2008. It is defined having 1 sign bit, 5 exponent bits and 10 mantissa bits.
- The exponent is biased by $2^{\text{expbits-1}}$ - 1 = 15, the mantissa is normalized and does not store the implicit bit. The canonical NaN as specified in the RiscV F standard extension for this format is the bit pattern 0x7e00.
- Vectors of narrower floating-point formats are packed inside the core registers. These packed narrower values within one core register are labelled as op 0..n and henceforth called operand entries. The first entry, op 0 is always located at the lowest portion of the register.
- The floating-point control and status register fcsr is used as defined in the RiscV F standard extension. The accrued exception flags are collected from all the parallel sub-operations and logic-ORed into their positions in the status register. Rounding modes for all parallel operations are driven with the same rounding mode from frm.

- The NaN generation and propagation scheme for vectorial formats is analogous to their counterparts defined in the respective scalar extensions and applies to all operand and result entrie independently. Where an invalid operation exception would be raised in the scalar instructions, it is triggered if at least one operation on entries of the vectorial instruction triggers the invalid operation condition.
- The replication bit R can be set for all instructions operating on at least two sources, rs1 and rs2. If set, the first entry op0 of rs2 is used for all entries of rs2. This scalar replication of op0 in rs2 can be used to perform vector-scalar operations. No bits in the actual register at rs2 are modified, unless it also acts as the destination rd. As such, VFADD.R.vfmt, VFSUB.R.vfmt, VFMUL.R.vfmt, VFDIV.R.vfmt perform floating-point addition, subtraction, multiplication, and division, respectively, between the entries of rs1 and the first entry in rs2, writing the result to rd. VFMIN.R.vfmt and VFMAX.R.vfmt write, respectively, the smaller or larger entries of rs1 and the first entry in rs2 to rd.
- All vectorial floating-point operations that perform rounding select the rounding mode using frm from the float control and status register fcsr. All entries are treated with the same rounding mode.
- VFLT.vfmt, VFLE.vfmt, VFGT.vfmt and VFGE.vfmt perform what the IEEE 754-2008 standard refers to as signaling comparisons: that is, an Invalid Operation exception is raised if either input is NaN. VFEQ.vfmt and VFNE.vfmt perform a quiet comparison: only signaling NaN inputs cause an Invalid Operation exception. For all six instructions, the result entry is 0 if either operand entry is NaN.
- VF-CLASS.vfmt, examine the entries in register rs1 and write to register rd a vector of 8-bit classification blocks that indicate the class of each floating-point entry using a bit mask. The format of the classification block is described in the table below. The corresponding mask bit in an entry of rd will be set if the property is true and clear otherwise. All other mask bits in rd are cleared. Note that exactly one mask bit in each entry rd will be set. The sign bit is placed in the highest bit of the block. The classification block corresponding to the first entry in rs1 is placed in the lowest byte of rd, the second inside the second byte and so on. For less populous formats, the classification block of the highest entry is replicated to fill the entire destination register.

B7: SignBit	B6: !SignBit	B5..B0: Mask	Meaning
0		000000	rs1 is $+\infty$
1		000000	rs1 is $-\infty$
0		000001	rs1 is a positive normal number
1		000001	rs1 is a negative normal number
0		000010	rs1 is a positive subnormal number
1		000010	rs1 is a negative subnormal number
0		000011	rs1 is $+0$
1		000011	rs1 is -0
–		000100	rs1 is a signaling NaN
–		000101	rs1 is a quiet NaN

31:30	29:25	24:20	19:15	14	13:12	11:7	6:0	
f2	**vecfltop**	**rs2**	**rs1**	**R**	**vfmt**	**rd**	**opcode**	**instruction**
10	00001	rs2	rs1	0	10	rd	0110011	vfadd.h rd, rs1, rs2
10	00001	rs2	rs1	1	10	rd	0110011	vfadd.r.h rd, rs1, rs2
10	00010	rs2	rs1	0	10	rd	0110011	vfsub.h rd, rs1, rs2
10	00010	rs2	rs1	1	10	rd	0110011	vfsub.r.h rd, rs1, rs2
10	00011	rs2	rs1	0	10	rd	0110011	vfmul.h rd, rs1, rs2
10	00011	rs2	rs1	1	10	rd	0110011	vfmul.r.h rd, rs1, rs2
10	00100	rs2	rs1	0	10	rd	0110011	vfdiv.h rd, rs1, rs2
10	00100	rs2	rs1	1	10	rd	0110011	vfdiv.r.h rd, rs1, rs2
10	00101	rs2	rs1	0	10	rd	0110011	vfmin.h rd, rs1, rs2
10	00101	rs2	rs1	1	10	rd	0110011	vfmin.r.h rd, rs1, rs2
10	00110	rs2	rs1	0	10	rd	0110011	vfmax.h rd, rs1, rs2
10	00110	rs2	rs1	1	10	rd	0110011	vfmax.r.h rd, rs1, rs2
10	00110	00000	rs1	0	10	rd	0110011	vfsqrt.h rd, rs1, rs2
10	01000	rs2	rs1	0	10	rd	0110011	vfmac.h rd, rs1, rs2
10	01000	rs2	rs1	1	10	rd	0110011	vfmac.r.h rd, rs1, rs2

31:30	29:25	24:20	19:15	14	13:12	11:7	6:0	
10	01001	rs2	rs1	0	10	rd	0110011	vfmre.h rd, rs1, rs2
10	01001	rs2	rs1	1	10	rd	0110011	vfmre.r.h rd, rs1, rs2
10	01100	00001	rs1	0	10	rd	0110011	vfclass.h rd, rs1
10	01101	rs2	rs1	0	10	rd	0110011	vfsgnj.h rd, rs1, rs2
10	01101	rs2	rs1	1	10	rd	0110011	vfsgnj.r.h rd, rs1, rs2
10	01110	rs2	rs1	0	10	rd	0110011	vfsgnjn.h rd, rs1, rs2
10	01110	rs2	rs1	1	10	rd	0110011	vfsgnjn.r.h rd, rs1, rs2
10	01111	rs2	rs1	0	10	rd	0110011	vfsgnjx.h rd, rs1, rs2
10	01111	rs2	rs1	1	10	rd	0110011	vfsgnjx.r.h rd, rs1, rs2
10	10000	rs2	rs1	0	10	rd	0110011	vfeq.h rd, rs1, rs2
10	10000	rs2	rs1	1	10	rd	0110011	vfeq.r.h rd, rs1, rs2
10	10001	rs2	rs1	0	10	rd	0110011	vfne.h rd, rs1, rs2
10	10001	rs2	rs1	1	10	rd	0110011	vfne.r.h rd, rs1, rs2
10	10010	rs2	rs1	0	10	rd	0110011	vflt.h rd, rs1, rs2
10	10010	rs2	rs1	1	10	rd	0110011	vflt.r.h rd, rs1, rs2
10	10011	rs2	rs1	0	10	rd	0110011	vfge.h rd, rs1, rs2
10	10011	rs2	rs1	1	10	rd	0110011	vfge.r.h rd, rs1, rs2
10	10100	rs2	rs1	0	10	rd	0110011	vfle.h rd, rs1, rs2
10	10100	rs2	rs1	1	10	rd	0110011	vfle.r.h rd, rs1, rs2
10	10101	rs2	rs1	0	10	rd	0110011	vfgt.h rd, rs1, rs2
10	10101	rs2	rs1	1	10	rd	0110011	vfgt.r.h rd, rs1, rs2
10	11000	rs2	rs1	0	10	rd	0110011	vfcpka.h.s rd, rs1, rs2
10	01100	00010	rs1	0	10	rd	0110011	vfcvt.x.h rd, rs1
10	01100	00010	rs1	1	10	rd	0110011	vfcvt.xu.h rd, rs1
10	01100	00011	rs1	0	10	rd	0110011	vfcvt.h.x rd, rs1
10	01100	00011	rs1	1	10	rd	0110011	vfcvt.h.xu rd, rs1

5.1.8.6 ISA extension. 16b non IEEE floating point instructions, vector

31:30	29:25	24:20	19:15	14	13:12	11:7	6:0	
f2	**vecfltop**	**rs2**	**rs1**	**R**	**vfmt**	**rd**	**opcode**	**instruction**
10	00001	rs2	rs1	0	01	rd	0110011	vfadd.ah rd,rs1,rs2
10	00001	rs2	rs1	1	01	rd	0110011	vfadd.r.ah rd,rs1,rs2
10	00010	rs2	rs1	0	01	rd	0110011	vfsub.ah rd,rs1,rs2
10	00010	rs2	rs1	1	01	rd	0110011	vfsub.r.ah rd,rs1,rs2
10	00011	rs2	rs1	0	01	rd	0110011	vfmul.ah rd,rs1,rs2
10	00011	rs2	rs1	1	01	rd	0110011	vfmul.r.ah rd,rs1,rs2
10	00100	rs2	rs1	0	01	rd	0110011	vfdiv.ah rd,rs1,rs2
10	00100	rs2	rs1	1	01	rd	0110011	vfdiv.r.ah rd,rs1,rs2
10	00101	rs2	rs1	0	01	rd	0110011	vfmin.ah rd,rs1,rs2
10	00101	rs2	rs1	1	01	rd	0110011	vfmin.r.ah rd,rs1,rs2
10	00110	rs2	rs1	0	01	rd	0110011	vfmax.ah rd,rs1,rs2
10	00110	rs2	rs1	1	01	rd	0110011	vfmax.r.ah rd,rs1,rs2
10	00110	0	rs1	0	01	rd	0110011	vfsqrt.ah rd,rs1,rs2
10	01000	rs2	rs1	0	01	rd	0110011	vfmac.ah rd,rs1,rs2
10	01000	rs2	rs1	1	01	rd	0110011	vfmac.r.ah rd,rs1,rs2
10	01001	rs2	rs1	0	01	rd	0110011	vfmre.ah rd,rs1,rs2
10	01001	rs2	rs1	1	01	rd	0110011	vfmre.r.ah rd,rs1,rs2
10	01100	1	rs1	0	01	rd	0110011	vfclass.ah rd,rs1
10	01101	rs2	rs1	0	01	rd	0110011	vfsgnj.ah rd,rs1,rs2
10	01101	rs2	rs1	1	01	rd	0110011	vfsgnj.r.ah rd,rs1,rs2
10	01110	rs2	rs1	0	01	rd	0110011	vfsgnjn.ah rd,rs1,rs2

31:30	29:25	24:20	19:15	14	13:12	11:7	6:0	
10	01110	rs2	rs1	1	01	rd	0110011	vfsgnjn.r.ah rd,rs1,rs2
10	01111	rs2	rs1	0	01	rd	0110011	vfsgnjx.ah rd,rs1,rs2
10	01111	rs2	rs1	1	01	rd	0110011	vfsgnjx.r.ah rd,rs1,rs2
10	10000	rs2	rs1	0	01	rd	0110011	vfeq.ah rd,rs1,rs2
10	10000	rs2	rs1	1	01	rd	0110011	vfeq.r.ah rd,rs1,rs2
10	10001	rs2	rs1	0	01	rd	0110011	vfne.ah rd,rs1,rs2
10	10001	rs2	rs1	1	01	rd	0110011	vfne.r.ah rd,rs1,rs2
10	10010	rs2	rs1	0	01	rd	0110011	vflt.ah rd,rs1,rs2
10	10010	rs2	rs1	1	01	rd	0110011	vflt.r.ah rd,rs1,rs2
10	10011	rs2	rs1	0	01	rd	0110011	vfge.ah rd,rs1,rs2
10	10011	rs2	rs1	1	01	rd	0110011	vfge.r.ah rd,rs1,rs2
10	10100	rs2	rs1	0	01	rd	0110011	vfle.ah rd,rs1,rs2
10	10100	rs2	rs1	1	01	rd	0110011	vfle.r.ah rd,rs1,rs2
10	10101	rs2	rs1	0	01	rd	0110011	vfgt.ah rd,rs1,rs2
10	10101	rs2	rs1	1	01	rd	0110011	vfgt.r.ah rd,rs1,rs2
10	11000	rs2	rs1	0	01	rd	0110011	vfcpka.ah.s rd,rs1,rs2
10	01100	10	rs1	0	01	rd	0110011	vfcvt.x.ah rd,rs1
10	01100	10	rs1	1	01	rd	0110011	vfcvt.xu.ah rd,rs1
10	01100	11	rs1	0	01	rd	0110011	vfcvt.ah.x rd,rs1
10	01100	11	rs1	1	01	rd	0110011	vfcvt.ah.xu rd,rs1

5.1.9 GAP9 interrupts and events

5.1.9.1 SoC events

Event number	Event name	Description
0	UDMA_CORE_LIN_0_EVT	uDMA linear addrgen0 transfer done
1	UDMA_CORE_LIN_1_EVT	uDMA linear addrgen1 transfer done
...	...	...
63	UDMA_CORE_LIN_63_EVT	uDMA linear addrgen63 transfer done
64	UDMA_CORE_2D_0_EVT	uDMA 2D addrgen0 transfer done
65	UDMA_CORE_2D_1_EVT	uDMA 2D addrgen1 transfer done
...	...	...
71	UDMA_CORE_2D_7_EVT	uDMA 2D addrgen7 transfer done
72	UDMA_CORE_CH2CH_0_EVT	uDMA Channel2Channel FIFO0 transfer done
73	UDMA_CORE_CH2CH_1_EVT	uDMA Channel2Channel FIFO1 transfer done
...	...	...
79	UDMA_CORE_CH2CH_7_EVT	uDMA Channel2Channel FIFO7 transfer done
80	SPI0_EVT	SPI0 End Of Transfer
81	SPI1_EVT	SPI1 End Of Transfer
82	SPI2_EVT	SPI2 End Of Transfer
83	SPI3_EVT	SPI3 End Of Transfer
84	UART0_RX_EVT	UART0 RX event
85	UART0_TX_EVT	UART1 TX event
86	UART0_ERR_EVT	UART2 ERR event
87	UART1_RX_EVT	UART0 RX event
88	UART1_TX_EVT	UART1 TX event
89	UART1_ERR_EVT	UART2 ERR event
90	UART2_RX_EVT	UART0 RX event
91	UART2_TX_EVT	UART1 TX event
92	UART2_ERR_EVT	UART2 ERR event

Event number	Event name	Description
93	UART3_RX_EVT	UART0 RX event
94	UART3_TX_EVT	UART1 TX event
95	UART3_ERR_EVT	UART2 ERR event
96	UART4_RX_EVT	UART0 RX event
97	UART4_TX_EVT	UART1 TX event
98	UART4_ERR_EVT	UART2 ERR event
99	I2C0_LEADER_MAIN_EVT	I2C0 main leader event
100	I2C0_FOLLOWER_MAIN_EVT	I2C0 main follower event
101	Reserved	Reserved
102	I2C1_LEADER_MAIN_EVT	I2C1 main leader event
103	I2C1_FOLLOWER_MAIN_EVT	I2C1 main follower event
104	Reserved	Reserved
105	I2C2_LEADER_MAIN_EVT	I2C2 main leader event
106	I2C2_FOLLOWER_MAIN_EVT	I2C2 main follower event
107	Reserved	Reserved
108	I2C3_LEADER_MAIN_EVT	I2C3 main leader event
109	I2C3_FOLLOWER_MAIN_EVT	I2C3 main follower event
110	Reserved	Reserved
111	HYPER0_EVT	Memory interface 0 event
112	HYPER1_EVT	Memory interface 1 event
113	MRAM_ERASE_EVT	MRAM erase done event
114	MRAM_TX_EVT	MRAM TX done event
115	MRAM_TRIM_EVT	MRAM trimming configuration done event
116	MRAM_RX_EVT	MRAM RX done event
117	FILTER_EOT_EVT	FILTER EOT event
118	FILTER_ACT_EVT	FILTER ACT event
119	TIMESTAMP	Timestamp event
120	FFC0_EVT	Floating point/fixed point converter event
121	FFC1_EVT	Floating point/fixed point converter event
122	FFC2_EVT	Floating point/fixed point converter event
123	FFC3_EVT	Floating point/fixed point converter event
124	SFU_EVT	SFU event
125-127	Reserved	Reserved
128	I3C0_EVT	I3C event
129	DIV_REF_FAST_CLOCK_-EVT	Raised at each clock rising edge of divided REF FAST clock
130	PMU_DLC_PICL_OK_EVT	PMU PICL bus OK
131	PMU_DLC_SCU_OK_EVT	PMU SCU transition OK
132	ADV_TIMER0_EVT	PWM event 0
133	ADV_TIMER1_EVT	PWM event 1
134	ADV_TIMER2_EVT	PWM event 2
135	ADV_TIMER3_EVT	PWM event 3
136	GPIO_EVT	GPIO event
137	RTC_APB_EVT	RTC APB transfer OK
138	RTC_EVT	RTC event
139	TIMEOUT_EVT_0	uDMA timeout event 0
140	TIMEOUT_EVT_1	uDMA timeout event 1
141	TIMEOUT_EVT_2	uDMA timeout event 2
142	TIMEOUT_EVT_3	uDMA timeout event 3
143	TIMEOUT_EVT_4	uDMA timeout event 4
144	TIMEOUT_EVT_5	uDMA timeout event 5
145	TIMEOUT_EVT_6	uDMA timeout event 6
146	TIMEOUT_EVT_7	uDMA timeout event 7

5.1.9.2 FC interrupts

IRQ number	IRQ name	Description
0	SW_IRQ_0	Software IRQ 0 triggered by the FC interrupt controller
1	SW_IRQ_1	Software IRQ 1 triggered by the FC interrupt controller
2	SW_IRQ_2	Software IRQ 2 triggered by the FC interrupt controller
3	SW_IRQ_3	Software IRQ 3 triggered by the FC interrupt controller
4	SW_IRQ_4	Software IRQ 4 triggered by the FC interrupt controller
5	SW_IRQ_5	Software IRQ 5 triggered by the FC interrupt controller
6	SW_IRQ_6	Software IRQ 6 triggered by the FC interrupt controller
7	FC_STATUS_IRQ	Peripheral status interrupt for FC
8	DMA_FC_IRQ	Cluster DMA interrupt
9	DECOMPR_DONE_IRQ	Cluster decompressor interrupt
10	FC_TIMER0_LO_IRQ	FC timer 0 low interrupt
11	FC_TIMER0_HI_IRQ	FC timer 0 high interrupt
12	FC_TIMER1_LO_IRQ	FC timer 1 low interrupt
13	FC_TIMER1_HI_IRQ	FC timer 1 high interrupt
14	SFU_IRQ	SFU event interrupt
15	GPIO_IRQ	GPIO event interrupt
16	REF_CLOCK_RISE_IRQ	Interrupt raised at each SoC slow clock rising edge
17	ADV_TIMER0_IRQ	PWM channel 0 interrupt
18	ADV_TIMER1_IRQ	PWM channel 1 interrupt
19	ADV_TIMER2_IRQ	PWM channel 2 interrupt
20	ADV_TIMER3_IRQ	PWM channel 3 interrupt
21	RTC_APB_IRQ	RTC APB transaction done interrupt
22	RTC_IRQ	RTC event interrupt
23	XIP_IRQ	XIP interrupt
24	DLC_PICL_OK_IRQ	PMU DLC PICL transaction done interrupt
25	DLC_SCU_OK_IRQ	PMU SCU sequence done interrupt
26	SOC_PERIPH_IRQ	SoC event FIFO not empty interrupt
27	QUIDDIKEY_IRQ	Quiddikey event interrupt
28	MPU_ERR_IRQ	MPU error/bad access interrupt
29	SOC_PERIPH_ERR_IRQ	SoC peripherals error interrupt
30	HYPER0_EOT_IRQ	Memory interface 0 end of transfer interrupt
31	HYPER1_EOT_IRQ	Memory interface 1 end of transfer interrupt

5.1.9.3 Cluster interrupts

IRQ number	IRQ name	Description
0	SW_IRQ_0	Software IRQ 0 triggered by the event unit
1	SW_IRQ_1	Software IRQ 1 triggered by the event unit
2	SW_IRQ_2	Software IRQ 2 triggered by the event unit
3	SW_IRQ_3	Software IRQ 3 triggered by the event unit
4	SW_IRQ_4	Software IRQ 4 triggered by the event unit
5	SW_IRQ_5	Software IRQ 5 triggered by the event unit
6	SW_IRQ_6	Software IRQ 6 triggered by the event unit
7	SW_IRQ_7	Software IRQ 7 triggered by the event unit
8	DMA_IRQ_0	DMA interrupt 0
9	DMA_IRQ_1	DMA interrupt 1
10	CL_TIMER_LO_IRQ	Cluster timer low interrupt
11	CL_TIMER_HI_IRQ	Cluster timer high interrupt
12	NE16_IRQ_0	NE16 interrupt 0
13	NE16_IRQ_1	NE16 interrupt 1

IRQ number	IRQ name	Description
14	Reserved	Reserved
15	Reserved	Reserved
16	BARRIER_IRQ	Barrier reached interrupt
17	MUTEX_IRQ	Mutex acquired interrupt
18	DISPATCH_IRQ	Dispatch FIFO value popped interrupt
19	Reserved	Reserved
20	Reserved	Reserved
21	Reserved	Reserved
22	CLUSTER_IRQ_0	Cluster event 0 interrupt
23	CLUSTER_IRQ_1	Cluster event 1 interrupt
24	DECOMP_IRQ	Decompressor done interrupt
25	Reserved	Reserved
26	Reserved	Reserved
27	SOC_PERIPH_IRQ	SoC event FIFO not empty interrupt
28	Reserved	Reserved
29	Reserved	Reserved
30	Reserved	Reserved
31	Reserved	Reserved

5.2 FC Peripherals

5.2.1 GAP9 ROM

This section describes the behavior of the GAP9 boot ROM, and in particular the content of GAP9 eFuse used by the ROM software. The value of these eFuses directly impacts the behavior of the chip during the boot process. Each eFuse bit is 0 by default, can be programmed to 1, but cannot be set back to 0 afterwards.

5.2.1.1 Features

Boot From Pads

Two *bootsel* pads (WAKEUP_SPI2_SDI and WAKEUP_SPI2_SCK) can be used to specify the chip boot mode, encoded on 2 bits (eFuse, JTAG, Hyperflash, MRAM). This mode is overwriting the boot mode specified in the eFuse. As these pins may also be used for other functions, for example the SPI wake-up, their monitoring for boot mode selection can be disabled by the BOOTMODE0_NOCHECK and BOOTMODE1_NOCHECK fields of the INFO_2 register.

Normal Boot Preemption

Using JTAG boot mode on bootsel pads will preempt the normal boot procedure described in the eFuse. This allows easy reprogramming of the flash, even though everything has be programmed in the eFuse. Only the bootmode is overwritten. This possibility of preempting the normal procedure can be disabled by deactivating the JTAG in the eFuse.

FLL Setup

The DCO0 of the FLL can be setup by the ROM to feed the SOC and PERIPH clocks before the binary is loaded from flash. The desired configuration is then described in the eFuse.

Pad Function Setting

The function of all pads can be setup by the ROM according to what is specified in the eFuse.

Flash Parameters

Several parameters for the flash can be given in the eFuse:

- Flash chip select.
- Flash interface.
- Hyperbus chip size, delay and latency.
- MRAM trim config.
- Periph clock divider, to tune the desired frequency on the interface.

Retentive Boot

The ROM will detect a wakeup from retentive state and will directly jump to the binary without reloading it.

XIP Boot

The ROM can handle both static binary loading and XIP configuration to load the part of the binary which is static and let the rest be fetched by XIP.

5.2.1.2 Boot Modes

The boot mode can be specified in several ways on the two bootsel pads. 0, 1 or 2 pads can be used to specify the bootmode on the pads. If 1 pad is used, it is used to determine bit 0 of the boot mode, and if 2 are used, bootsel pad 0 (WAKEUP_SPI2_SCK) is used for bit 0 and bootsel pad 1 (WAKEUP_SPI2_SDI) is used for bit 1 of the bootmode, with the following meaning:

- b00: The bootmode is specified in the eFuses.
- b01: The bootmode will be specified in the JTAG confreg register. This will stop the ROM, which will poll the confreg registers until a bootmode is specified.
- b10: The ROM will boot from HYPER flash.
- b11: The ROM will boot from MRAM.

The bootmode described in the JTAG confreg and in the eFuses is using the following format (see following sections for more details on each mode):

- 0: JTAG stop.
- 1: Hyperflash boot.
- 2: SPI flash boot.
- 3: MRAM boot.
- 4: SPI slave boot.

JTAG Bootmode

When the bootsel pads are b01, the ROM will stop the normal boot procedure and will enter a polling loop on the JTAG confreg to wait for a bootmode. As soon as the ROM sees bit0 to 1, it will shift the confreg by 1 bit on the right and will execute the corresponding bootmode, which can be either "JTAG stop" for letting the external loader taking control over the boot procedure, or any other mode, for example to boot from flash.

JTAG Stop

This bootmode can be used to tell the ROM to stop any activity and put the system in a state where an external loader can take over the boot procedure, for example for loading a binary through JTAG.

When the ROM detects this mode, it will acknowledge it in the JTAG confreg register, as soon as the system is ready, so that the external JTAG driver can know when it can take over the boot. Then the ROM will continue polling the JTAG confreg register, waiting for another boot mode, for example to allow the external loader to ask for an hyperflash boot. When the ROM continues with another boot mode sent through the JTAG confreg, it will read all the configuration from the eFuses like for the normal procedure, except for the bootmode.

Hyperflash Boot

The ROM will load the binary from hyperflash and jump to its entry point.

SPI Boot

The ROM will load the binary from SPI flash and jump to its entry point.

MRAM Boot

The ROM will load the binary from MRAM and jump to its entry point.

SPI Slave Boot

The ROM will enter a loop executing commands sent by an external loader to read and write memory and execute an entry point.

5.2.1.3 Oscillator Management

The ROM can be told through eFuse to setup the oscillators when booting. There are one setting for cold boot boot and one setting for non-retentive wakeup. The ROM won't do anything in retentive wakeup since it's just jump to the runtime. After wakeup, setting the oscillator can also be disabled dynamically from the runtime using the retention register.

A wait loop can be activated through eFuse so that the ROM waits for fast oscillator convergence. In this case it will wait until its frequency gets stable by comparing the number of oscillations during a fixed period, based on FLL open-loop oscillations. The wait loop have different settings between cold boot and wait-up since the oscillator may have been kept ON during deep sleep. The wait-loop for the wakeup can be controlled both with an eFuse and dynamically from the runtime using the retention register.

5.2.1.4 Wait Loops

The ROM contains several wait loops to respect the constraints of the HW. For example it may need to wait for a while after a flash has been reset before using it.

All the wait loops are using the fabric controller timer but the trigger source of the timer can be specified in field TIMER_SOURCE of eFuse INFO_1 with the following possible values:

- 0: the timer is triggered by the FC FLL. Note that the FLL can be either used in open loop (which is the default if the FLL eFuses are not configured) or in FLL close loop.
- 1: The timer is triggered by the slow 32k oscillator.
- 2: The timer is triggered by the fast oscillator.

By default, when eFuses are not setup, the timer will work in FLL open-loop mode and will compute default wait loops based on the worst-case at 50MHz, which means all the wait loops will actually be longer than expected, since the real FLL frequency will be below and will vary.

A wait loop of 0 cycle always means the wait loop should be dropped.

Be careful that once the FLL configuration or the source of the timer are given through eFuses, all the wait loops become invalid and must all be given through eFuses, taking into account the given configuration and the fact that all wait loops are expressed in number of timer cycles in the eFuses.

5.2.1.5 MRAM Support

Features

The ROM is able to load the boot binary from MRAM. It can also configure the XIP so that the binary is partially or entirely executed directly from MRAM through XIP.

Before using the MRAM, the rom powers it up at PMU level, configure the uDMA, does the internal power-up and send the trim config.

The PMU power-up is bypassed if the bit 2 of field REBOOT of APB SoC Controller register SLEEP_CTRL is set to 1, so that it can support the PMU sequence where the MRAM is kept ON. In this case, the ROM will still do the internal power-up and send the trim config.

When XIP support is enabled, the uDMA MRAM periph is kept enabled and uDMA address generator 0 is used for RX channel so that the loaded binary can use XIP addresses immediately. If XIP support is not enabled, the uDMA MRAM periph is disabled but the MRAM is kept on once the boot process is done.

EFuse Configuration

The TRIM configuration is taken from eFuses if field MRAM_TRIM of eFuse INFO_2 is set to 1. In this case it is specified through MRAM_TRIM_SIZE and all the eFuses starting at MRAM_TRIM_START. MRAM_TRIM_SIZE gives the size of the trim config in 32bit words and also the number of eFuses containing the config. The ROM will start reading the config from eFuse MRAM_TRIM_START and will continue with the following eFuses until the specified size is read. The ROM supports at most 32 eFuses for the trim config. If no trim config is given, the following one is taken (recommended settings from the specifications):

```
0x00000000
0xFD798D00
0x620490D0
0x0406082E
0x0580001B
0x00000000
0x00000000
```

The uDMA divider is taken from field CLKDIV of eFuse INFO_2 if field CLKDIV_SETUP of eFuse INFO_2 is set to 1. The divider is applied on the peripheral clock. Otherwise a default divider of 5 is taken so that the MRAM can be used in FLL open-loop when the FLL is not configured by the eFuses.

During the MRAM power-up, RSTB must be kept low during a minimal amount of time. This time is specified for cold boot in eFuse MRAM_RESET_WAIT_CYCLES if field MRAM_RESET_WAIT of eFuse INFO_1 is set to 1, and for wake-up boot in eFuse WAKE_MRAM_RESET_WAIT_CYCLES if field WAKE_MRAM_-RESET_WAIT of eFuse INFO_2 is set to 1. When one of them is not specified, a default value of 60µs is taken. Please refer to the section about wait loops above.

After the trim config has been sent, the ROM must wait for a while before the MRAM can be accessed. This time is specified in eFuse FLASH_WAIT if field FLASH_WAIT of eFuse INFO_3 is set to 1. If it is not specified, a default value of 20µs is taken. Please refer to the section about wait loops above.

5.2.1.6 Hyperbus/Octospi Support

Features

The ROM is able to load the boot binary from an external hyperflash or an ATXP032 octospi flash. Other octospi flashs may be supported if the specifications are slightly different from the ATXP032 specifications. It can also configure the XIP so that the binary is partially or entirely executed directly from flash through XIP.

When XIP support is enabled, the uDMA hyperbus periph is kept enabled and uDMA address generator 64 is used for RX channel so that the loaded binary can use XIP addresses immediately. If XIP support is not enabled, the uDMA hyperbus periph is disabled once the boot process is done.

The flash configuration phase can be customized.

For octospi flash the following operations are possible in this order:

- Flash reset through a GPIO pulse with flash reset wait loop
- Flash wakeup with flash wakeup wait loop
- Flash status write with spi conf wait loop
- Flash custom commands
- Flash wait loop

For Hyperflash the following operations are possible in this order:

- Flash wakeup with flash wakeup wait loop
- Flash reset through a GPIO pulse with flash reset wait loop
- Flash reset through a command with flash reset wait loop
- Flash init
- Flash custom commands
- Flash wait loop

EFuse Configuration

The hyperbus/octospi interface where the flash is connected is taken from field FLASH_ITF of eFuse INFO_3 if field FLASH_ITF_SETUP of eFuse INFO_3 is set to 1. The same for the chip select with fields FLASH_CS and FLAS_CS_SETUP of eFuse INFO_3. If both interface and chip select are not specified, the ROM assumes the flash is connected on interface 0 and chip select 0. In this case if field PADFUN0_SETUP of eFuse INFO_1 is set to 0, it will also configure the padframe so that the flash can be accessed.

The uDMA divider is taken from field CLKDIV of eFuse INFO_2 if field CLKDIV_SETUP of eFuse INFO_2 is set to 1. The divider is applied on the peripheral clock. Otherwise a default divider of 0 is taken so that the flash can be used in FLL open-loop when the FLL is not configured by the eFuses.

The hyperbus/octospi delay setup during configuration is taken from field HYPER_DELAY of eFuse INFO_3 if field HYPER_DELAY_SETUP of eFuse INFO_3 is set to 1. Otherwise a default delay of 1 is taken.

The hyperbus/octospi latency setup during configuration is taken from field HYPER_LATENCY of eFuse INFO_3 if field HYPER_LATENCY_SETUP of eFuse INFO_3 is set to 1. Otherwise a default latency of 6 is taken.

The hyperbus/octospi address used to access the base of the device connected to chip select 1 is taken from eFuse FLASH_OFFSET if field FLASH_OFFSET_SETUP of eFuse INFO_3 is set to 1. Otherwise a default value of 0x10000000 is taken.

When a flash reset is generated, the ROM will apply the wait loop specified in eFuse FLASH_RESET_WAIT if field FLASH_RESET_WAIT of eFuse INFO_3 is set to 1. Please refer to the section about wait loops above.

If field FLASH_WAKEUP of eFuse INFO_3 is set to 1, the ROM will generate a flash wakeup command after non-retentive wakeup.

A wait loop specified in eFuse FLASH_WAKEUP_WAIT is applied after flash wake-up if field FLASH_WAKEUP_WAIT of eFuse INFO_3 is set to 1. Otherwise a default wait loop of 300µs is applied.

The ROM can apply 4 custom flash commands, either after cold boot and/or after non-retentive wakeup. Each command specified in eFuse FLASH_CMD_* is applied after cold boot if field FLASH_CMD_* of eFuse INFO_3 is set to 1 and after non-retentive wakeup if field FLASH_CMD_*_DS of eFuse INFO_3 is set to 1.

A final wait loop specified in eFuse FLASH_WAIT is applied before the flash is accessed if field FLASH_WAIT of eFuse INFO_3 is set to 1.

A GPIO pulse is generated to reset the flash if field FLASH_GPIO_PULSE_GEN of eFuse INFO_4 is set to 1. In this case, the ID of the GPIO is taken from field FLASH_GPIO_PULSE_ID of eFuse INFO_4. The polarity of the pulse is taken from field FLASH_GPIO_PULSE_POL of eFuse INFO_4.

When a GPIO pulse is generated to reset the flash, the duration of the pulse is taken from eFuse FLASH_GPIO_PULSE_WAIT if field FLASH_GPIO_PULSE_WAIT of eFuse INFO_4 is set to 1. Otherwise the duration is 100µs.

Octospi EFuse Configuration

After the flash status register has been setup, the ROM will apply the wait loop specified in eFuse SPI_CONF_WAIT_CYCLES if field SPI_CONF_WAIT of eFuse INFO_2 is set to 1. Otherwise it will wait for 10us. Please refer to the section about wait loops above.

The ROM will write to the status register the value specified in eFuse FLASH_STATUS if field FLASH_STATUS_SET of eFuse INFO_2 is set to 2, set it to 0x1b880200 if the field is 0, which corresponds to octospi DTR mode with all sectors unprotected, and will do nothing if the field is set to 1.

For read accesses, the ROM will use the latency specified in field FLASH_LATENCY_VALUE of eFuse INFO_2 if field FLASH_LATENCY_SET of eFuse INFO_2 is set to 1. Otherwise it will use 22.

The ROM will use the flash commands specified in eFuse FLASH_COMMANDS if field FLASH_COMMANDS_SET of eFuse INFO_2 is set to 1. Otherwise it will use 0x06 for write enable, 0x71 for write status and 0x0B for read operation.

Hyperbus EFuse Configuration

A flash reset is generated if field FLASH_RESET of eFuse INFO_3 is set to 1.

A flash init sequence is sent if field FLASH_INIT of eFuse INFO_3 is set to 1.

5.2.1.7 SPI Slave Support

Features

The ROM is able to receive commands from an external device through SPI to read and write memory and execute an entry point.

For that, a protocol is defined to let the external loader interact with the ROM. The external loader should send requests of the following form:

```
typedef struct
{
    uint8_t frame_start;     // Magic number for start of frame (0x5A)
    uint8_t cmd;             // Command to be executed
```

(continues on next page)

(continued from previous page)

```
    uint32_t address;        // Address when needed by the command
    uint8_t size_minus_1;    // Size minus 1 when needed by the command
    uint8_t unused;
} __attribute__((packed)) spi_boot_req_t;
```

The ROM will reply in full duplex with a one byte packet containing the special value 0xA5. As soon as the external loader detects it, it should go further with the command. The command can be either 1 for a read, 2 for a write and 3 for a jump to an address.

For a read command, the external loader should wait for the special value 0x79. As soon as it detects it, the next byte should be ignored and read the value right after.

For a write command, a second buffer containing the special value 0x5A, a dummy byte and all the bytes to be written should be sent. The ROM will sent back the special value 0xA5 in full duplex.

For a jump command, the address specifies the address where the ROM should jump.

EFuse Configuration

The SPI interface where the external loader is connected is taken from field FLASH_ITF of eFuse INFO_3 if field FLASH_ITF_SETUP of eFuse INFO_3 is set to 1. Otherwise the ROM assumes it is connected on interface 2.

5.2.1.8 Secured Boot

Binary Encryption and Authentification

The ROM only supports binary encryptiosizen if the binary is authenticated by the quiddikey block.

The enroll operation of the quiddikey (to generate the Activation Code), is supposed to have been done once (in factory). The AC has to be stored somewhere. This AC can then be used to wrap a key using the quiddikey to produce the Key Code. The AC is stored in the header binary to be loaded by the ROM while the KC is stored in an eFuse.

When booting, the ROM first loads the binary header which is never encrypted. If the binary is encrypted, it loads the AC from the header and the KC from the eFuse and do the quiddikey unwrap with them. The boot process goes on only if this operation is successfull. It then configures the AES, and then proceeds with normal section loading from flash, AES + Hyper will manage the decryption.

Finaly it checks the eFuse, and potentially blocks quiddikey enroll and unwrap operations.

The key can either be generated offline, or in the chip. If it is generated offline, the fuser has to be executed to compute the Key Code using the quiddikey and store it in the eFuse. If it is generated in the chip, the flasher will produce the flash image unencrypted together with a table of sections to be encrypted. Both will be given to the flasher which will use the table to dynamically encrypt the sections when they are sent to the flasher.

Hash

A hash can also be given in each section description. In this case, the ROM gets the hash after decryption of the section descriptor, then decrypts the section, computes the hash of the decrypted section, compares with the expected one, and aborts the execution if they are different.

Secured Boot After Wake-Up

After a retentive wakeup, the ROM does not do anything with quiddikey and AES, but just jumps into the binary kept in L2 with retention. The binary is then supposed to unwrap again the quiddikey. This is a costly operation (55000 cycles), which is needed for full secure key. It is also possible for the binary to store the key in L2 and reload it manually to the AES to avoid this cost, but in this case the key must be know by the SW part.

After a non-retentive wakeup, the unwrap operation has been unblocked by the HW after wakeup, which will allow the ROM to redo the quiddikey configuration.

Boot With No AES and No Quiddikey

In this case, the key is stored in the eFuse and the decryption is done in SW by the ROM.

5.2.1.9 Register map for reserved eFuses

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
INFO_1	0x0	32	Information eFuse 1
INFO_2	0x1	32	Information eFuse 2
INFO_3	0x2	32	Information eFuse 3
FLL_DRR	0x3	32	FLL DRR value (only used when FLL_GLOBAL_SETUP is 1)
FLL_CCR1_PRE_-LOCK	0x4	32	FLL CCR1 value set before locking the FLL (only used when FLL_GLOBAL_SETUP is 1)
FLL_CCR1_POST_-LOCK	0x5	32	FLL CCR1 value set after locking the FLL (only used when FLL_GLOBAL_SETUP is 1)
FLL_CCR2	0x6	32	FLL CCR2 value (only used when FLL_GLOBAL_SETUP is 1)
FLL_F0CR1	0x7	32	FLL F0CR1 value (only used when FLL_GLOBAL_SETUP is 1)
FLL_F0CR2	0x8	32	FLL F0CR2 value (only used when FLL_GLOBAL_SETUP is 1)
PADFUN0	0x9	32	PADFUN0 value (only used when PADFUN0_SETUP is 1)
PADFUN1	0xA	32	PADFUN1 value (only used when PADFUN1_SETUP is 1)
PADFUN2	0xB	32	PADFUN2 value (only used when PADFUN2_SETUP is 1)
PADFUN3	0xC	32	PADFUN3 value (only used when PADFUN3_SETUP is 1)
PADFUN4	0xD	32	PADFUN4 value (only used when PADFUN4_SETUP is 1)
PADFUN5	0xE	32	PADFUN5 value (only used when PADFUN5_SETUP is 1)
FEATURE_DISABLE	0xF	32	Specify the list of features which must be deactivated by the ROM boot
SECURE_BOOT	0x10	32	Specify AES configuration
AES_KEY0	0x11	32	Word 0 of AES key
AES_KEY1	0x12	32	Word 1 of AES key
AES_KEY2	0x13	32	Word 2 of AES key
AES_KEY3	0x14	32	Word 3 of AES key
AES_KEY4	0x15	32	Word 4 of AES key
AES_KEY5	0x16	32	Word 5 of AES key
AES_KEY6	0x17	32	Word 6 of AES key
AES_KEY7	0x18	32	Word 7 of AES key
FEATURE_DISABLE_-QK	0x19	32	Specify list of QuiddiKey features which must be disabled by the ROM boot

Name	Offset	Width	Description
WAIT_XTAL_PERIOD	0x20	32	When WAIT_XTAL is 1, this gives the timer period at which the oscillator is checked
WAIT_XTAL_DELTA	0x21	32	When WAIT_XTAL is 1, this gives the delta under which the oscillator is considered stable
WAIT_XTAL_MIN	0x22	32	When WAIT_XTAL is 1, this gives the number of stable checks after which the wait is considered successful
WAIT_XTAL_MAX	0x23	32	When WAIT_XTAL is 1, this gives the number of unstable checks after which the wait is considered failing and is aborted
REF_CLK_WAIT_CYCLES	0x24	32	When REF_CLK_WAIT is 1, this gives the number of clock cycles after which the ROM can start accessing the pads after cold boot (used clock is selected by the TIMER_SOURCE field of INFO_1 eFuse)
REF_CLK_WAIT_CYCLES_DEEP_SLEEP	0x25	32	When REF_CLK_WAIT_DEEP_SLEEP is 1, this gives the number of clock cycles after which the ROM can start accessing the pads after non-retentive wakeup (used clock is selected by the TIMER_SOURCE field of INFO_1 eFuse)
FAST_CLK_DIV_POW2	0x26	32	When FAST_CLK_DIV_POW2_SETUP is 1, the ROM will setup the fast clock divider with this value (the real division factor is the power of two of this value)
WAKEUP_FLL_DRR	0x27	32	Wakeup FLL DRR value (only used when FLL_GLOBAL_SETUP is 1)
WAKEUP_FLL_CCR1_PRE_LOCK	0x28	32	Wakeup FLL CCR1 value set before locking the FLL (only used when FLL_GLOBAL_SETUP is 1)
WAKEUP_FLL_CCR1_POST_LOCK	0x29	32	Wakeup FLL CCR1 value set after locking the FLL (only used when FLL_GLOBAL_SETUP is 1)
WAKEUP_FLL_CCR2	0x2A	32	Wakeup FLL CCR2 value (only used when FLL_GLOBAL_SETUP is 1)
WAKEUP_FLL_F0CR1	0x2B	32	Wakeup FLL F0CR1 value (only used when FLL_DCO0_SETUP is 1)
WAKEUP_FLL_F0CR2	0x2C	32	Wakeup FLL F0CR2 value (only used when FLL_DCO0_SETUP is 1)
WAKE_FAST_CLK_DIV_POW2	0x2D	32	When WAKE_FAST_CLK_DIV_POW2_SETUP is 1, the ROM will setup the fast clock divider with this value after non-retentive deep sleep (the real division factor is the power of two of this value)
MRAM_RESET_WAIT_CYCLES	0x2E	32	Number of cycles to wait after MRAM has been reset (this is a number of cycles for the timer, whatever the timer source is)
WAKE_MRAM_RESET_WAIT_CYCLES	0x2F	32	Number of cycles to wait after MRAM has been reset after a non-retentive wakeup (this is a number of cycles for the timer, whatever the timer source is)
SPI_CONF_WAIT_CYCLES	0x30	32	Number of cycles to wait after the spiflash has been configured (this is a number of cycles for the timer, whatever the timer source is)
FLASH_OFFSET	0x31	32	Flash offset
FLL_WAIT_CYCLES	0x32	32	Number of cycles to wait before the FLL is configured (this is a number of cycles for the timer, whatever the timer source is)
FLL_WAKE_WAIT_CYCLES	0x33	32	Number of cycles to wait before the FLL is configured after non-retentive wakeup (this is a number of cycles for the timer, whatever the timer source is)
FLASH_RESET_WAIT	0x35	32	Wait loop after flash reset
FLASH_CMD_1	0x36	32	First additionnal custom command
FLASH_CMD_2	0x37	32	Second additionnal custom command
FLASH_CMD_3	0x38	32	Third additionnal custom command
FLASH_CMD_4	0x39	32	Fourth additionnal custom command

Name	Offset	Width	Description
FLASH_WAIT	0x3A	32	Apply a wait loop before using the flash
FLASH_WAKEUP_-WAIT	0x3B	32	Wait loop when waiting for flash wakup
FLASH_STATUS	0x3C	32	Flash status register value
FLASH_COMMANDS	0x3D	32	Flash commands
INFO_4	0x3E	32	Information eFuse 4
FLASH_GPIO_-PULSE_WAIT	0x3F	32	Number of cycles the ROM should wait after it has set the GPIO to active
NEVA_CFG	0x40	32	Number of cycles the ROM should wait after it has set the GPIO to active
MRAM_TRIM_SIZE	0x41	32	When MRAM_TRIM is 1, this gives the size of the MRAM trim config
MRAM_TRIM_START	0x42	32	When MRAM_TRIM is 1, this is the first eFuse storing the MRAM trim configuration

INFO_1

Information eFuse 1

Bit #	R/W	Name	Reset	Description
2:0	R/W	PLATFORM	0x0	Platform on which the execution is being done. This is only used for test purpose on simulation platform and should be kept to 0 on real platform. Possible values: 0: Undefined, 1: FPGA, 2: RTL, 3: GVSOC, 4: BOARD
10:3	R/W	BOOTMODE	0x0	Bootmode that the ROM should follow (see bootmode section for more details). Possible values: 0: JTAG stop, 1: Hyperflash boot, 2: SPI flash boot, 3: MRAM boot, 4: SPI slave boot
11	R/W	ENCRYPTED	0x0	1 if the binary to be loaded from flash is encrypted.
12	R/W	WAIT_XTAL	0x0	1 if the ROM should wait for stabilization of the oscillator.
13	R/W	ICACHE_ENABLED	0x0	1 if the ROM should activate FC icache.
14	R/W	FLL_GLOBAL_SETUP	0x0	1 if the ROM should configure FLL global registers (drr, ccr1 and ccr2).
15	R/W	FLL_DCO0_SETUP	0x0	1 if the ROM should configure DCO 0. (f0cr1 and f0cr2)
16	R/W	PADFUN0_SETUP	0x0	1 if the ROM should configure PADFUN0.
17	R/W	PADFUN1_SETUP	0x0	1 if the ROM should configure PADFUN1.
18	R/W	PADFUN2_SETUP	0x0	1 if the ROM should configure PADFUN2.
19	R/W	PADFUN3_SETUP	0x0	1 if the ROM should configure PADFUN3.
20	R/W	PADFUN4_SETUP	0x0	1 if the ROM should configure PADFUN4.
21	R/W	PADFUN5_SETUP	0x0	1 if the ROM should configure PADFUN5.
22	R/W	PMU_WAIT_RESET_SKIP	0x0	1 if the ROM should not wait for end of reset sequence.
24:23	R/W	TIMER_SOURCE	0x0	Clock source for the timer used for generating wait loops: 0: FLL, 1: 32kHz reference clock, 2: divided fast clock.
25	R/W	FAST_CLK_DIV_POW2_SETUP	0x0	1 if the ROM should setup the fast clock divider with the content of FAST_CLK_DIV_POW2.
26	R/W	OSC_CTRL_SETUP	0x0	1 if the ROM should setup the oscillator control register with the content of OSC_CTRL.
29:27	R/W	OSC_CTRL	0x0	Content of oscillator control register when it is setup.
30	R/W	FEATURE_DISABLE_SET	0x0	Set feature disable register from what is specified in FEATURE_DISABLE.
31	R/W	MRAM_RESET_WAIT	0x0	Set number of cycles to wait after the MRAM has been reset. The number of cycles is taken from MRAM_RESET_WAIT_CYCLES.

INFO_2

Information eFuse 2

Bit #	R/W	Name	Reset	Description
0	R/W	CLKDIV_SETUP	0x0	1 if the ROM should take the peripheral divider from field CLKDIV of eFuse INFO_2. If it is 0, a default divider of 0 is taken for Hyper flash and SPI flash, and a divider of 2 for MRAM.
5:1	R/W	CLKDIV	0x0	Peripheral divider. 0 or 1 do not divide, other values divide by the specified value.
6	R/W	JTAG_LOCK	0x0	1 if the ROM should not authorize JTAG accesses.
7	R/W	REF_CLK_WAIT	0x0	1 if the ROM should wait before accessing the pads. The duration of the wait is the number of ref clock cycles described in eFuse REF_CLK_WAIT_CYCLES.
8	R/W	REF_CLK_WAIT_DEEP_SLEEP	0x0	1 if the ROM should wait before accessing the pads after non-retentive wakeup. The duration of the wait is the number of ref clock cycles described in eFuse REF_CLK_WAIT_CYCLES_DEEP_SLEEP.
9	R/W	BOOTMODE0_NOCHECK	0x0	1 if the ROM should not use bootsel pad 0 for choosing boot mode.
10	R/W	BOOTMODE1_NOCHECK	0x0	1 if the ROM should not use bootsel pad 1 for choosing boot mode.
11	R/W	MRAM_TRIM	0x0	1 if the ROM should configure MRAM trim before using the MRAM.
12	R/W	WAKE_FAST_CLK_DIV_POW2_SETUP	0x0	1 if the ROM should setup the fast clock divider with the content of WAKE_FAST_CLK_DIV_POW2 after non-retentive deep sleep.
13	R/W	WAKE_OSC_CTRL_SETUP	0x0	1 if the ROM should setup the oscillator control register with the content of WAKE_OSC_CTRL after non-retentive deep sleep.
15:14	R/W	WAKE_OSC_CTRL	0x0	Content of oscillator control register when it is setup after non-retentive deep sleep.
16	R/W	SPI_CONF_WAIT	0x0	Set number of cycles to wait after the spiflash has been configured. The number of cycles is taken from SPI_CONF_WAIT_CYCLES.
17	R/W	WAKE_WAIT_XTAL	0x0	1 if the ROM should wait for stabilization of the oscillator after non-retentive wakeup.
18	R/W	FLL_WAIT	0x0	1 if the ROM should wait before configuring the FLL. The number of cycles is taken from FLL_WAIT_CYCLES.
19	R/W	FLL_WAKE_WAIT	0x0	1 if the ROM should wait before configuring the FLL. The number of cycles is taken from FLL_WAKE_WAIT_CYCLES.
22:21	R/W	FLASH_STATUS_SET	0x0	0 if the ROM should set the flash status register to a default value, 1, if it should do nothing or 2 if it should apply the status found in FLASH_STATUS.
23	R/W	FLASH_COMMANDS_SET	0x0	1 if the ROM should take flash commands from FLASH_COMMANDS.
24	R/W	FLASH_LATENCY_SET	0x0	1 if the ROM should take flash latency from FLASH_LATENCY_VALUE.
29:25	R/W	FLASH_LATENCY_VALUE	0x0	Flash latency.
30	R/W	WAKE_MRAM_RESET_WAIT	0x0	Set number of cycles to wait after the MRAM has been reset after non-retentive wakeup. The number of cycles is taken from WAKE_MRAM_RESET_WAITC_CYCLES.

INFO_3

Information eFuse 3

Bit #	R/W	Name	Reset	Description
0	R/W	FLASH_CS_SETUP	0x0	Setup Chip Select of the flash to be used for the binary loading.
1	R/W	FLASH_CS	0x0	Chip Select of the flash to be used for the binary loading.
2	R/W	FLASH_ITF_SETUP	0x0	Setup interface ID where the flash is connected for the binary loading.
4:3	R/W	FLASH_ITF	0x0	Interface ID where the flash is connected for the binary loading.
5	R/W	FLASH_OFFSET_-SETUP	0x0	Set the offset of the flash. Offset is given in FLASH_OFFSET.
6	R/W	HYPER_DELAY_-SETUP	0x0	Set Hyperbus delay.
9:7	R/W	HYPER_DELAY	0x0	Hyperbus delay.
10	R/W	HYPER_LATENCY_-SETUP	0x0	Set Hyperbus latency.
15:11	R/W	HYPER_LATENCY	0x0	Hyperbus latency.
16	R/W	RESERVED	0x0	–
17	R/W	FLASH_WAKEUP	0x0	Wakeup the flash after non-retentive deep sleep wakeup.
18	R/W	FLASH_RESET	0x0	Reset the flash before using it.
19	R/W	FLASH_INIT	0x0	Init the flash before using it.
20	R/W	FLASH_WAIT	0x0	Apply a wait loop before using the flash.
21	R/W	FLASH_CMD_1	0x0	First additionnal custom command.
22	R/W	FLASH_CMD_2	0x0	Second additionnal custom command.
23	R/W	FLASH_CMD_3	0x0	Third additionnal custom command.
24	R/W	FLASH_CMD_4	0x0	Fourth additionnal custom command.
25	R/W	FLASH_CMD_1_DS	0x0	First additionnal custom command after non-retentive wakeup.
26	R/W	FLASH_CMD_2_DS	0x0	Second additionnal custom command after non-retentive wakeup.
27	R/W	FLASH_CMD_3_DS	0x0	Third additionnal custom command after non-retentive wakeup.
28	R/W	FLASH_CMD_4_DS	0x0	Fourth additionnal custom command after non-retentive wakeup.
29	R/W	FLASH_RESET_WAIT	0x0	Wait loop after flash reset.
30	R/W	FLASH_WAKEUP_-WAIT	0x0	Wait loop when waiting for flas wakeup.

FLL_DRR

FLL DRR value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLL_CCR1_PRE_LOCK

FLL CCR1 value set before locking the FLL (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLL_CCR1_POST_LOCK

FLL CCR1 value set after locking the FLL (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLL_CCR2

FLL CCR2 value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLL_F0CR1

FLL F0CR1 value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLL_F0CR2

FLL F0CR2 value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN0

PADFUN0 value (only used when PADFUN0_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN1

PADFUN1 value (only used when PADFUN1_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN2

PADFUN2 value (only used when PADFUN2_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN3

PADFUN3 value (only used when PADFUN3_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN4

PADFUN4 value (only used when PADFUN4_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

PADFUN5

PADFUN5 value (only used when PADFUN5_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FEATURE_DISABLE

Specify the list of features which must be deactivated by the ROM boot

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

SECURE_BOOT

Specify AES configuration

Bit #	R/W	Name	Reset	Description
0	R/W	SECURE_ONLY	0x0	Only allow secure boot.
1	R/W	AES_QK	0x0	Use QK as key source.
2	R/W	AES_USER	0x0	Use eFuse as key source.
3	R/W	AES_USER_KEY_-SIZE	0x0	Key size for "user" security (256=1,128=0).
4	R/W	CRC_EN	0x0	Enable crc check.
5	R/W	KEY_LOCK	0x0	Lock AES key in eFuse.
6	R/W	SIGN_ONLY	0x0	Sign only mode, no encryption.
7	R/W	QK_LOCK	0x0	Lock QK features.

AES_KEY0

Word 0 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY1

Word 1 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY2

Word 2 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY3

Word 3 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY4

Word 4 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY5

Word 5 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY6

Word 6 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

AES_KEY7

Word 7 of AES key

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FEATURE_DISABLE_QK

Specify list of QuiddiKey features which must be disabled by the ROM boot

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAIT_XTAL_PERIOD

When WAIT_XTAL is 1, this gives the timer period at which the oscillator is checked

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

WAIT_XTAL_DELTA

When WAIT_XTAL is 1, this gives the delta under which the oscillator is considered stable

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

WAIT_XTAL_MIN

When WAIT_XTAL is 1, this gives the number of stable checks after which the wait is considered successful

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

WAIT_XTAL_MAX

When WAIT_XTAL is 1, this gives the number of unstable checks after which the wait is considered failing and is aborted

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

REF_CLK_WAIT_CYCLES

When REF_CLK_WAIT is 1, this gives the number of clock cycles after which the ROM can start accessing the pads after cold boot (used clock is selected by the TIMER_SOURCE field of INFO_1 eFuse)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

REF_CLK_WAIT_CYCLES_DEEP_SLEEP

When REF_CLK_WAIT_DEEP_SLEEP is 1, this gives the number of clock cycles after which the ROM can start accessing the pads after non-retentive wakeup (used clock is selected by the TIMER_SOURCE field of INFO_1 eFuse)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FAST_CLK_DIV_POW2

When FAST_CLK_DIV_POW2_SETUP is 1, the ROM will setup the fast clock divider with this value (the real division factor is the power of two of this value)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_DRR

Wakeup FLL DRR value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_CCR1_PRE_LOCK

Wakeup FLL CCR1 value set before locking the FLL (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_CCR1_POST_LOCK

Wakeup FLL CCR1 value set after locking the FLL (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_CCR2

Wakeup FLL CCR2 value (only used when FLL_GLOBAL_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_F0CR1

Wakeup FLL F0CR1 value (only used when FLL_DCO0_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKEUP_FLL_F0CR2

Wakeup FLL F0CR2 value (only used when FLL_DCO0_SETUP is 1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

WAKE_FAST_CLK_DIV_POW2

When WAKE_FAST_CLK_DIV_POW2_SETUP is 1, the ROM will setup the fast clock divider with this value after non-retentive deep sleep (the real division factor is the power of two of this value)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

MRAM_RESET_WAIT_CYCLES

Number of cycles to wait after MRAM has been reset (this is a number of cycles for the timer, whatever the timer source is)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

WAKE_MRAM_RESET_WAIT_CYCLES

Number of cycles to wait after MRAM has been reset after a non-retentive wakeup (this is a number of cycles for the timer, whatever the timer source is)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

SPI_CONF_WAIT_CYCLES

Number of cycles to wait after the spiflash has been configured (this is a number of cycles for the timer, whatever the timer source is)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FLASH_OFFSET

Flash offset

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FLL_WAIT_CYCLES

Number of cycles to wait before the FLL is configured (this is a number of cycles for the timer, whatever the timer source is)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FLL_WAKE_WAIT_CYCLES

Number of cycles to wait before the FLL is configured after non-retentive wakeup (this is a number of cycles for the timer, whatever the timer source is)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FLASH_RESET_WAIT

Wait loop after flash reset

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

FLASH_CMD_1

First additionnal custom command

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_CMD_2

Second additionnal custom command

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_CMD_3

Third additionnal custom command

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_CMD_4

Fourth additionnal custom command

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_WAIT

Apply a wait loop before using the flash

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_WAKEUP_WAIT

Wait loop when waiting for flash wakup

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_STATUS

Flash status register value

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

FLASH_COMMANDS

Flash commands

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

INFO_4

Information eFuse 4

Bit #	R/W	Name	Reset	Description
0	R/W	FLASH_GPIO_-PULSE_GEN	0x0	Generate a pulse on a GPIO before using the flash.
1	R/W	FLASH_GPIO_-PULSE_WAIT	0x0	1 if the ROM should wait after it has set the GPIO to active.
2	R/W	FLASH_GPIO_-PULSE_POL	0x0	1 if the pulse should be active high.
9:3	R/W	FLASH_GPIO_-PULSE_ID	0x0	GPIO pulse ID.
10	R/W	NEVA_CFG	0x0	Set the NEVA configuration from NEVA_CFG.

FLASH_GPIO_PULSE_WAIT

Number of cycles the ROM should wait after it has set the GPIO to active

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

NEVA_CFG

Number of cycles the ROM should wait after it has set the GPIO to active

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Value to copy to register

MRAM_TRIM_SIZE

When MRAM_TRIM is 1, this gives the size of the MRAM trim config

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

MRAM_TRIM_START

When MRAM_TRIM is 1, this is the first eFuse storing the MRAM trim configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x0	Configuration value to be used by boot code

5.2.2 FLL

See clocking architecture section for more details on FLL use.

5.2.2.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
FSR	0x0	32	FLL status register
DRR	0x4	32	DCO Range register
TTR	0x8	32	Temperature Tracking Register
F0CR1	0xC	32	FLL0 Configuration Register 1
F0CR2	0x10	32	FLL0 Configuration Register 2
F1CR1	0x14	32	FLL1 Configuration Register 1
F1CR2	0x18	32	FLL1 Configuration Register 2
F2CR1	0x1C	32	FLL2 Configuration Register 1
F2CR2	0x20	32	FLL2 Configuration Register 2
F3CR1	0x24	32	FLL3 Configuration Register 1
F3CR2	0x28	32	FLL3 Configuration Register 2
CCR1	0x2C	32	Clock Configuration register 1
CCR2	0x30	32	Clock Configuration register 2

FSR

FLL status register

Bit #	R/W	Name	Reset	Description
0	R	LOCK0	0x0	LOCK0. Lock status for Feedback clock 0.
1	R	CLMP_LO_ERR0	0x0	Clamp error low for FLL0. Set when new DCO value for FLL0 < DCO_MIN value.
2	R	CLMP_HI_ERR0	0x0	Clamp error high for FLL0. Set when new DCO value for FLL0 > DCO_MAX value.
3	R	FDC_SAT_ERR0	0x0	FDC saturation error for FLL0. Set when FDC counter of FLL0 overflows.
4	R	LOCK1	0x0	LOCK1. Lock status for Feedback clock 1.
5	R	CLMP_LO_ERR1	0x0	Clamp error low for FLL1. Set when new DCO value for FLL1 < DCO_MIN value.
6	R	CLMP_HI_ERR1	0x0	Clamp error high for FLL1. Set when new DCO value for FLL1 > DCO_MAX value.
7	R	FDC_SAT_ERR1	0x0	FDC saturation error for FLL1. Set when FDC counter of FLL1 overflows.
8	R	LOCK2	0x0	LOCK2. Lock status for Feedback clock 3.
9	R	CLMP_LO_ERR2	0x0	Clamp error low for FLL2. Set when new DCO value for FLL2 < DCO_MIN value.
10	R	CLMP_HI_ERR2	0x0	Clamp error high for FLL2. Set when new DCO value for FLL2 > DCO_MAX value.
11	R	FDC_SAT_ERR2	0x0	FDC saturation error for FLL2. Set when FDC counter of FLL2 overflows.
12	R	LOCK3	0x0	LOCK3. Lock status for Feedback clock 3.
13	R	CLMP_LO_ERR3	0x0	Clamp error low for FLL3. Set when new DCO value for FLL3 < DCO_MIN value.
14	R	CLMP_HI_ERR3	0x0	Clamp error high for FLL3. Set when new DCO value for FLL3 > DCO_MAX value.
15	R	FDC_SAT_ERR3	0x0	FDC saturation error for FLL3. Set when FDC counter of FLL3 overflows.
24:16	R	DCOD	0x00F	Current DCO input code of the selected FLL (see DCOD_SEL field of DRR register).

DRR

DCO Range register

Bit #	R/W	Name	Reset	Description
8:0	R/W	DCOD_MIN	0x000	Minimum value allowed for DCO code.
24:16	R/W	DCOD_MAX	0x086	Maximum value allowed for DCO code.
29:28	R/W	DCOD_SEL	0x0	Selection of the FLL the DCO of which can be read back through FSR register (b00: FLL0, b01: FLL1, b10: FLL2, b11: FLL3).

TTR

Temperature Tracking Register

Bit #	R/W	Name	Reset	Description
23:0	R/W	REFRESH	0x000000	Number of ref clock cycles between two integration periods.

F0CR1

FLL0 Configuration Register 1

Bit #	R/W	Name	Reset	Description
0	R/W	DCO_EN	0x1	DCO enable for FLL0 (active high). 0: DCO0 is disabled – FBKCLK0 is inactive. 1: DCO0 is enabled – FBKCLK0 is managed according to the FLL0 configuration (default state).
1	R/W	OP_MODE	0x0	FLL0 operating mode. 0: open loop mode (default state). 1: closed loop mode.
2	R/W	TTM_EN	0x0	FLL0 temperature tracking mode enable. In open loop mode: do not care. In closed loop mode: 0: the frequency is always regulated (at each integration period). 1: the frequency is regulated at a rate controlled by REFRESH parameter of the TTR register.
3	R/W	DITH_EN	0x0	FLL0 dithering enable. In open loop mode: do not care. In closed loop mode: when set, enable dithering pattern generator.
7:4	R/W	LOOP_GAIN	0x7	FLL0 loop gain setting. Default: $2^{-7} = 1/256$.
15:8	R/W	LOCK_TOL	0x0A	FLL0 lock tolerance. Margin around the target multiplication factor per integration period (MFI) within which the output clock is considered stable (i.e. the clock is stable when target $\|\text{MFI} - N_{\text{FDC}}\| <$ LOCK_TOL.
25:16	R/W	ITG_PER	0x003	FLL0 integration period. Defines the duration of one integration period i.e. the number of REFCLK cycles during which the FDC counter is enabled. Integration period duration = (ITG_PER + 1) REFCLK cycles
31:26	R/W	STBL	0x10	FLL0 stable/unstable clock cycles before asserting/deasserting LOCK0. In closed loop mode, if LOCK0=0 (resp. 1): number of integration periods during which FBKCLK0 is stable (resp. unstable) before LOCK0 is asserted (resp. deasserted). In open loop mode, number of FBKCLK0 cycles until LOCK0 is asserted.

F0CR2

FLL0 Configuration Register 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	MFI	0x0001	Target clock multiplication factor per integration period for FLL0 (closed loop mode).
24:16	R/W	DCOD_OLM	0x00F	DCO input code for FLL0 (open loop mode).

F1CR1

FLL1 Configuration Register 1

Bit #	R/W	Name	Reset	Description
0	R/W	DCO_EN	0x0	DCO enable for FLL1 (active high). 0: DCO1 is disabled (default state) – FBKCLK1 is inactive. 1: DCO1 is enabled – FBKCLK1 is managed according to the FLL1 configuration.
1	R/W	OP_MODE	0x0	FLL1 operating mode. 0: open loop mode (default state). 1: closed loop mode.
2	R/W	TTM_EN	0x0	FLL1 temperature tracking mode enable. In open loop mode: do not care. In closed loop mode: 0: the frequency is always regulated (at each integration period). 1: the frequency is regulated at a rate controlled by REFRESH parameter of the TTR register.
3	R/W	DITH_EN	0x0	FLL1 dithering enable. In open loop mode: do not care. In closed loop mode: when set, enable dithering pattern generator.
7:4	R/W	LOOP_GAIN	0x7	FLL1 loop gain setting. Default: $2^{-7} = 1/256$.
15:8	R/W	LOCK_TOL	0x0A	FLL1 lock tolerance. Margin around the target multiplication factor per integration period (MFI) within which the output clock is considered stable (i.e. the clock is stable when target $\|\text{MFI} - N_{\text{FDC}}\| < \text{LOCK\_TOL}$.
25:16	R/W	ITG_PER	0x003	FLL1 integration period. Defines the duration of one integration period i.e. the number of REFCLK cycles during which the FDC counter is enabled. Integration period duration = (ITG_PER + 1) REFCLK cycles
31:26	R/W	STBL	0x10	FLL1 stable/unstable clock cycles before asserting/deasserting LOCK1. In closed loop mode, if LOCK1=0 (resp. 1): number of integration periods during which FBKCLK1 is stable (resp. unstable) before LOCK1 is asserted (resp. deasserted). In open loop mode, number of FBKCLK1 cycles until LOCK1 is asserted.

F1CR2

FLL1 Configuration Register 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	MFI	0x0001	Target clock multiplication factor per integration period for FLL1 (closed loop mode).
24:16	R/W	DCOD_OLM	0x00F	DCO input code for FLL1 (open loop mode).

F2CR1

FLL2 Configuration Register 1

Bit #	R/W	Name	Reset	Description
0	R/W	DCO_EN	0x0	DCO enable for FLL2 (active high). 0: DCO2 is disabled (default state) – FBKCLK2 is inactive. 1: DCO2 is enabled – FBKCLK2 is managed according to the FLL2 configuration.
1	R/W	OP_MODE	0x0	FLL2 operating mode. 0: open loop mode (default state). 1: closed loop mode.
2	R/W	TTM_EN	0x0	FLL2 temperature tracking mode enable. In open loop mode: do not care. In closed loop mode: 0: the frequency is always regulated (at each integration period). 1: the frequency is regulated at a rate controlled by REFRESH parameter of the TTR register.
3	R/W	DITH_EN	0x0	FLL2 dithering enable. In open loop mode: do not care. In closed loop mode: when set, enable dithering pattern generator.
7:4	R/W	LOOP_GAIN	0x7	FLL2 loop gain setting. Default: $2^{-7} = 1/256$.
15:8	R/W	LOCK_TOL	0x0A	FLL2 lock tolerance. Margin around the target multiplication factor per integration period (MFI) within which the output clock is considered stable (i.e. the clock is stable when target $\|\text{MFI} - N_{\text{FDC}}\| < \text{LOCK\_TOL}$.
25:16	R/W	ITG_PER	0x003	FLL2 integration period. Defines the duration of one integration period i.e. the number of REFCLK cycles during which the FDC counter is enabled. Integration period duration = (ITG_PER + 1) REFCLK cycles
31:26	R/W	STBL	0x10	FLL2 stable/unstable clock cycles before asserting/deasserting LOCK2. In closed loop mode, if LOCK2=0 (resp. 1): number of integration periods during which FBKCLK2 is stable (resp. unstable) before LOCK2 is asserted (resp. deasserted). In open loop mode, number of FBKCLK2 cycles until LOCK2 is asserted.

F2CR2

FLL2 Configuration Register 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	MFI	0x0001	Target clock multiplication factor per integration period for FLL2 (closed loop mode).
24:16	R/W	DCOD_OLM	0x00F	DCO input code for FLL2 (open loop mode).

F3CR1

FLL3 Configuration Register 1

Bit #	R/W	Name	Reset	Description
0	R/W	DCO_EN	0x0	DCO enable for FLL3 (active high). 0: DCO3 is disabled (default state) – FBKCLK3 is inactive. 1: DCO3 is enabled – FBKCLK3 is managed according to the FLL3 configuration.
1	R/W	OP_MODE	0x0	FLL3 operating mode. 0: open loop mode (default state). 1: closed loop mode.
2	R/W	TTM_EN	0x0	FLL3 temperature tracking mode enable. In open loop mode: do not care. In closed loop mode: 0: the frequency is always regulated (at each integration period). 1: the frequency is regulated at a rate controlled by REFRESH parameter of the TTR register.
3	R/W	DITH_EN	0x0	FLL3 dithering enable. In open loop mode: do not care. In closed loop mode: when set, enable dithering pattern generator.
7:4	R/W	LOOP_GAIN	0x7	FLL2 loop gain setting. Default: $2^{-7} = 1/256$.
15:8	R/W	LOCK_TOL	0x0A	FLL3 lock tolerance. Margin around the target multiplication factor per integration period (MFI) within which the output clock is considered stable (i.e. the clock is stable when target $\|\text{MFI} - N_{\text{FDC}}\| < \text{LOCK\_TOL}$.
25:16	R/W	ITG_PER	0x003	FLL3 integration period. Defines the duration of one integration period i.e. the number of REFCLK cycles during which the FDC counter is enabled. Integration period duration = (ITG_PER + 1) REFCLK cycles
31:26	R/W	STBL	0x10	FLL3 stable/unstable clock cycles before asserting/deasserting LOCK3. In closed loop mode, if LOCK3=0 (resp. 1): number of integration periods during which FBKCLK3 is stable (resp. unstable) before LOCK3 is asserted (resp. deasserted). In open loop mode, number of FBKCLK3 cycles until LOCK3 is asserted.

F3CR2

FLL3 Configuration Register 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	MFI	0x0001	Target clock multiplication factor per integration period for FLL3 (closed loop mode).
24:16	R/W	DCOD_OLM	0x00F	DCO input code for FLL3 (open loop mode).

CCR1

Clock Configuration register 1

Bit #	R/W	Name	Reset	Description
7:0	R/W	CLK0_DIV	0x02	Clock divider setting for output clock 0 (PERIPH clock); divider is bypassed if CLK0_DIV = 0 or 1.
15:8	R/W	CLK1_DIV	0x02	Clock divider setting for output clock 1 (SOC clock); divider is bypassed if CLK1_DIV = 0 or 1.
23:16	R/W	CLK2_DIV	0x00	Clock divider setting for output clock 2 (CLUSTER clock); divider is bypassed if CLK2_DIV = 0 or 1.
31:24	R/W	CLK3_DIV	0x00	Clock divider setting for output clock 3 (SFU clock); divider is bypassed if CLK3_DIV = 0 or 1.

CCR2

Clock Configuration register 2

Bit #	R/W	Name	Reset	Description
0	R/W	CLK0_SEL	0x1	Clock source selection for output clock 0 (PERIPH clock): 0: Ref clock (default). 1: FLL0 clock.
5:4	R/W	CLK1_SEL	0x1	Clock source selection for output clock 1 (SOC clock): 00: Ref clock. 01: FLL0 clock (default). 1x: FLL1 clock.
9:8	R/W	CLK2_SEL	0x0	Clock source selection for output clock 2 (CLUSTER clock): 00: Ref clock (default). 01: FLL0 clock. 10: FLL1 clock. 11: FLL2 clock.
14:12	R/W	CLK3_SEL	0x0	Clock source selection for output clock 3 (SFU clock): 000: Ref clock (default). 001: FLL0 clock. 010: FLL1 clock. 011: FLL2 clock. 1xx: FLL3 clock.
16	R/W	CKG0	0x1	FLL0 clock gated. 0: FLL0 is not gated. 1: FLL0 is clock gated by LOCK0 signal.
17	R/W	CKG1	0x1	FLL1 clock gated. 0: FLL1 is not gated. 1: FLL1 is clock gated by LOCK1 signal.
18	R/W	CKG2	0x1	FLL2 clock gated. 0: FLL2 is not gated. 1: FLL2 is clock gated by LOCK2 signal.
19	R/W	CKG3	0x1	FLL3 clock gated. 0: FLL3 is not gated. 1: FLL3 is clock gated by LOCK3 signal.

5.2.3 GPIO

GAP9 general purpose inputs/outputs (GPIOs) provide a general purpose control of digital input/output pins of the chip.

5.2.3.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
PADDIR_00_31	0x0	32	GPIO pad direction configuration register for GPIOs 0 to 31.
GPIOEN_00_31	0x4	32	GPIO enable register for GPIOs 0 to 31.
PADIN_00_31	0x8	32	GPIO pad input value register for GPIOs 0 to 31.
PADOUT_00_31	0xC	32	GPIO pad output value register for GPIOs 0 to 31.
PADOUTSET_00_31	0x10	32	GPIO pad output set register for GPIOs 0 to 31.
PADOUTCLR_00_31	0x14	32	GPIO pad output clear register for GPIOs 0 to 31.
INTEN_00_31	0x18	32	GPIO pad interrupt enable configuration register for GPIOs 0 to 31.
INTTYPE_00_15	0x1C	32	GPIO pad interrupt type for GPIOs 0 to 15.
INTTYPE_16_31	0x20	32	GPIO pad interrupt type for GPIOs 16 to 31.
INTSTATUS_00_31	0x24	32	GPIO pad interrupt status register for GPIOs 0 to 31.
PADDIR_32_63	0x48	32	GPIO pad direction configuration register for GPIOs 32 to 63.
GPIOEN_32_63	0x4C	32	GPIO enable register for GPIOs 32 to 63.
PADIN_32_63	0x50	32	GPIO pad input value register for GPIOs 32 to 63.
PADOUT_32_63	0x54	32	GPIO pad output value register for GPIOs 32 to 63.
PADOUTSET_32_63	0x58	32	GPIO pad output set register for GPIOs 32 to 63.
PADOUTCLR_32_63	0x5C	32	GPIO pad output clear register for GPIOs 32 to 63.
INTEN_32_63	0x60	32	GPIO pad interrupt enable configuration register for GPIOs 32 to 63.
INTTYPE_32_47	0x64	32	GPIO pad interrupt type for GPIOs 32 to 47.
INTTYPE_48_63	0x68	32	GPIO pad interrupt type for GPIOs 48 to 63.
INTSTATUS_32_63	0x6C	32	GPIO pad interrupt status register for GPIOs 32 to 63.
PADDIR_64_89	0x90	32	GPIO pad direction configuration register for GPIO 64 to 89.
GPIOEN_64_89	0x94	32	GPIO enable register for GPIO 64 to 89.
PADIN_64_89	0x98	32	GPIO pad input value register for GPIO 64 to 89.
PADOUT_64_89	0x9C	32	GPIO pad output value register for GPIO 64 to 89.
PADOUTSET_64_89	0xA0	32	GPIO pad output set register for GPIO 64 to 89.
PADOUTCLR_64_89	0xA4	32	GPIO pad output clear register for GPIO 64 to 89.
INTEN_64_89	0xA8	32	GPIO pad interrupt enable configuration register for GPIO 64 to 89.
INTTYPE_64_79	0xAC	32	GPIO pad interrupt type for GPIO 64 to 79.
INTTYPE_80_89	0xB0	32	GPIO pad interrupt type for GPIO 80 to 89.
INTSTATUS_64_89	0xB4	32	GPIO pad interrupt status register for GPIO 64 to 89.

PADDIR_00_31

GPIO pad direction configuration register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R/W	PADDIR	0x0	GPIO direction: bit i=0: GPIO i in input mode; bit i=1: GPIO i in output mode.

GPIOEN_00_31

GPIO enable register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R/W	GPIOEN	0x0	GPIO clock enable: bit i=0: disable clock for GPIO i; bit i=1: enable clock for GPIO i. GPIOs are grouped by 4, the clock gating of one group is done only if all 4 GPIOs' clocks are disable. Clock must be enabled for a GPIO to be used as an input.

PADIN_00_31

GPIO pad input value register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R	PADIN	0x0	GPIO input data: bit i corresponds to input data of GPIO i.

PADOUT_00_31

GPIO pad output value register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R/W	PADOUT	0x0	GPIO output data: bit i corresponds to input data of GPIO i.

PADOUTSET_00_31

GPIO pad output set register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	W	PADOUTSET	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO i output to 1.

PADOUTCLR_00_31

GPIO pad output clear register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	W	PADOUTCLR	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO i output to 0.

INTEN_00_31

GPIO pad interrupt enable configuration register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTEN	0x0	GPIO interrupt enable: bit i=0: disable interrupt for GPIO i; bit i=1: enable interrupt for GPIO i.

INTTYPE_00_15

GPIO pad interrupt type for GPIOs 0 to 15.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO i, type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTTYPE_16_31

GPIO pad interrupt type for GPIOs 16 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO (16+i), type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTSTATUS_00_31

GPIO pad interrupt status register for GPIOs 0 to 31.

Bit #	R/W	Name	Reset	Description
31:0	R	INTSTATUS	0x0	GPIO interrupt status flag. When read, bit i=1 signals that an interrupt has been received on GPIO i. Reading INTSTATUS clears its value and clears interrupt line.

PADDIR_32_63

GPIO pad direction configuration register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R/W	PADDIR	0x0	GPIO direction: bit i=0: GPIO (32+i) in input mode; bit i=1: GPIO (32+i) in output mode.

GPIOEN_32_63

GPIO enable register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R/W	GPIOEN	0x0	GPIO clock enable: bit i=0: disable clock for GPIO (32+i); bit i=1: enable clock for GPIO (32+i). GPIOs are grouped by 4, the clock gating of one group is done only if all 4 GPIOs' clocks are disable. Clock must be enabled for a GPIO to be used as an input.

PADIN_32_63

GPIO pad input value register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R	PADIN	0x0	GPIO input data: bit i corresponds to input data of GPIO (32+i).

PADOUT_32_63

GPIO pad output value register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R/W	PADOUT	0x0	GPIO output data: bit i corresponds to input data of GPIO (32+i).

PADOUTSET_32_63

GPIO pad output set register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	W	PADOUTSET	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO (32+i) output to 1.

PADOUTCLR_32_63

GPIO pad output clear register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	W	PADOUTCLR	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO (32+i) output to 0.

INTEN_32_63

GPIO pad interrupt enable configuration register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTEN	0x0	GPIO interrupt enable: bit i=0: disable interrupt for GPIO (32+i); bit i=1: enable interrupt for GPIO (32+i).

INTTYPE_32_47

GPIO pad interrupt type for GPIOs 32 to 47.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO (32+i), type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTTYPE_48_63

GPIO pad interrupt type for GPIOs 48 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO (48+i), type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTSTATUS_32_63

GPIO pad interrupt status register for GPIOs 32 to 63.

Bit #	R/W	Name	Reset	Description
31:0	R	INTSTATUS	0x0	GPIO interrupt status flag. When read, bit i=1 signals that an interrupt has been received on GPIO (32+i). Reading INTSTATUS clears its value and clears interrupt line.

PADDIR_64_89

GPIO pad direction configuration register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R/W	PADDIR	0x0	GPIO direction: bit i=0: GPIO (64+i) in input mode; bit i=1: GPIO (64+i) in output mode.

GPIOEN_64_89

GPIO enable register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R/W	GPIOEN	0x0	GPIO clock enable: bit i=0: disable clock for GPIO (64+i); bit i=1: enable clock for GPIO (64+i). GPIOs are grouped by 4, the clock gating of one group is done only if all 4 GPIOs' clocks are disable. Clock must be enabled for a GPIO to be used as an input.

PADIN_64_89

GPIO pad input value register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R	PADIN	0x0	GPIO input data: bit i corresponds to input data of GPIO (64+i).

PADOUT_64_89

GPIO pad output value register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R/W	PADOUT	0x0	GPIO output data: bit i corresponds to input data of GPIO (64+i).

PADOUTSET_64_89

GPIO pad output set register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	W	PADOUTSET	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO (64+i) output to 1.

PADOUTCLR_64_89

GPIO pad output clear register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	W	PADOUTCLR	0x0	GPIO output set bitfield: writing 1 to bit i sets GPIO (64+i) output to 0.

INTEN_64_89

GPIO pad interrupt enable configuration register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R/W	INTEN	0x0	GPIO interrupt enable: bit i=0: disable interrupt for GPIO (64+i); bit i=1: enable interrupt for GPIO (64+i).

INTTYPE_64_79

GPIO pad interrupt type for GPIO 64 to 79.

Bit #	R/W	Name	Reset	Description
31:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO (64+i), type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTTYPE_80_89

GPIO pad interrupt type for GPIO 80 to 89.

Bit #	R/W	Name	Reset	Description
19:0	R/W	INTTYPE	0x0	GPIO interrupt type (2 bits per GPIO): for GPIO (80+i), type is defined by bits (2xi+1, 2xi). If b00: interrupt on falling edge; if b01: interrupt on rising edge; if b10: interrupt on rising and falling edge; b11: Reserved.

INTSTATUS_64_89

GPIO pad interrupt status register for GPIO 64 to 89.

Bit #	R/W	Name	Reset	Description
25:0	R	INTSTATUS	0x0	GPIO interrupt status flag. When read, bit i=1 signals that an interrupt has been received on GPIO (64+i). Reading INTSTATUS clears its value and clears interrupt line.

5.2.4 SoC Controller

This block controls a number of global features of GAP9. SoC Controller registers starting from the 0x100 offset, i.e. above address 0x1A104100, are *always-on* registers: they keep their values when GAP9 enters and exits deep sleep.

5.2.4.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
INFO	0x0	32	Core information register
FC_BOOT	0x4	32	Boot address
FC_FETCH	0x8	32	FC Fetch enable
CL_ISOLATE	0xC	32	Isolate cluster register
PADFUN0	0x10	32	Pad mux config register (pads 0 to 15)
PADFUN1	0x14	32	Pad mux config register (pads 16 to 31)
PADFUN2	0x18	32	Pad mux config register (pads 32 to 47)
PADFUN3	0x1C	32	Pad mux config register (pads 48 to 63)
PADFUN4	0x20	32	Pad mux config register (pads 64 to 79)
PADFUN5	0x24	32	Pad mux config register (pads 80 to 89)
FEATURE_DISABLE_-QK	0x28	32	Feature disablement for Quiddikey features
PADCFG0	0x30	32	Pad function register (pads 0 to 3)
PADCFG1	0x34	32	Pad function register (pads 4 to 7)
PADCFG2	0x38	32	Pad function register (pads 8 to 11)
PADCFG3	0x3C	32	Pad function register (pads 12 to 15)
PADCFG4	0x40	32	Pad function register (pads 16 to 19)
PADCFG5	0x44	32	Pad function register (pads 20 to 23)
PADCFG6	0x48	32	Pad function register (pads 24 to 27)
PADCFG7	0x4C	32	Pad function register (pads 28 to 31)
PADCFG8	0x50	32	Pad function register (pads 32 to 35)
PADCFG9	0x54	32	Pad function register (pads 36 to 39)
PADCFG10	0x58	32	Pad function register (pads 40 to 43)
PADCFG11	0x5C	32	Pad function register (pads 44 to 47)
PADCFG12	0x60	32	Pad function register (pads 48 to 51)
PADCFG13	0x64	32	Pad function register (pads 52 to 55)
PADCFG14	0x68	32	Pad function register (pads 56 to 59)
PADCFG15	0x6C	32	Pad function register (pads 60 to 63)
PADCFG16	0x70	32	Pad function register (pads 64 to 67)
PADCFG17	0x74	32	Pad function register (pads 68 to 71)
PADCFG18	0x78	32	Pad function register (pads 72 to 75)
PADCFG19	0x7C	32	Pad function register (pads 76 to 79)
PADCFG20	0x80	32	Pad function register (pads 80 to 83)
PADCFG21	0x84	32	Pad function register (pads 84 to 87)
PADCFG22	0x88	32	Pad function register (pads 88 to 89)
REG_REPROG_PAD0	0x90	32	Controls reprogrammable pads 27, 28, 29, 30 and 34
REG_REPROG_PAD1	0x94	32	Controls reprogrammable pads 35, 40, 41, 42 and 43
REG_REPROG_PAD2	0x98	32	Controls reprogrammable pads 44, 45, 60, 61 and 62
REG_REPROG_PAD3	0x9C	32	Controls reprogrammable pads 63, 65, 66, 67 and 68
CL_BUSY	0xB0	32	Cluster busy register
JTAGREG	0xB4	32	JTAG external register

Name	Offset	Width	Description
REF_FAST_CLK_DIV	0xB8	32	Read only, reference fast clk divided by power of 2
SW_RST	0xBC	32	Software reset, reboot
CORESTATUS	0xC0	32	EOC and chip status register
BOOTSEL	0xC4	32	Value of bootsel pads
WD_RST_RST	0xC8	32	Rearm WD timeout
WD_RST_SET	0xCC	32	Set WD timer
RWM_CSI2	0xD0	32	Margin adjust settings for CSI-2 Controller
IDLE_MODE	0xD4	32	Activates IDLE MODE
RWM_ANC	0xD8	32	Margin adjust settings for SFU
REF_CLK_MUX	0xDC	32	Reference clock selection
SUPERVISOR_DBG	0xE0	32	
DBG_CTRL	0xE4	32	Debug access control
CLK_DIV_I3C	0xF0	32	Clock divider for I3C
CLK_EN_QUIDDIKEY	0xF4	32	Quiddikey enablement
SLEEP_CTRL_INFO	0xF8	32	Read-only access to sleep status
VERSION	0xFC	32	Show chip version (User controlled)
SLEEP_SPIS_CTRL	0x100	32	(*Always-on register*) Sleep SPIS control
SLEEP_CTRL	0x104	32	(*Always-on register*) Sleep control
SLEEP_GPIO_CTRL	0x108	32	(*Always-on register*) Sleep GPIO control
SLEEP_CNT_CTRL	0x10C	32	(*Always-on register*) Sleep counter control
REG_OSC_CTRL	0x110	32	(*Always-on register*) Controls fast oscillator
CLK_DIV_REF_-FAST_POW2	0x118	32	(*Always-on register*) Controls fast oscillator pow2 divider
FEATURE_DISABLE	0x120	32	(*Always-on register*) Feature disablement from always on (safe) domain
SLEEP_PAD_CFG0	0x140	32	(*Always-on register*) Pad control during sleep (pads 33 to 36)
SLEEP_PAD_CFG1	0x144	32	(*Always-on register*) Pad control during sleep (pads 37 to 40)
SLEEP_PAD_CFG2	0x148	32	(*Always-on register*) Pad control during sleep (pads 41 to 43 and 65)
SLEEP_PAD_CFG3	0x14C	32	(*Always-on register*) Pad control during sleep (pads 66 to 69)
SLEEP_PAD_CFG4	0x150	32	(*Always-on register*) Pad control during sleep (pads 81 to 84)
SLEEP_PAD_CFG5	0x154	32	(*Always-on register*) Pad control during sleep (pads 85 to 88)
SLEEP_PAD_CFG6	0x158	32	(*Always-on register*) Pad control during sleep (pad 89)
L2_CTRL_ACTIVE	0x15C	32	(*Always-on register*) Controls L2 power when SOC is powered on
L2_PWR_ACTIVE	0x160	32	(*Always-on register*) Controls L2 power when SOC is powered on
NEVACFG	0x164	32	(*Always-on register*) NEVA config
TRCCFG	0x168	32	(*Always-on register*) TRC config
RWM_L2_MEM	0x16C	32	(*Always-on register*) Read/write margins for L2 and ROM memories
CLU_SW_RSTN	0x170	32	(*Always-on register*) Cluster software reset
L2_PWR	0x174	32	(*Always-on register*) Controls L2 power when SOC is off (deep/retentive sleep)
L2_CTRL	0x178	32	(*Always-on register*) Controls L2 power when SOC is off (deep/retentive sleep)
RARMODE	0x184	32	(*Always-on register*) Controls configuration of the DC-DC modulation at low loads
ABBCFG	0x188	32	(*Always-on register*) Used to disable adaptive body-bias
L2_ACK	0x18C	32	(*Always-on register*) Acknowledge/status signals from L2 memories

INFO

Core information register

Bit #	R/W	Name	Reset	Description
15:0	R	NB_CL	0x0008	Number of clusters
31:16	R	NB_CORES	0x0001	Number of cores

FC_BOOT

Boot address

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDR	0x1A000080	FC Boot Address

FC_FETCH

FC Fetch enable

Bit #	R/W	Name	Reset	Description
0	R/W	FC_FE	0x1	FC Fetch Enable

CL_ISOLATE

Isolate cluster register

Bit #	R/W	Name	Reset	Description
0	R/W	EN	0x1	Isolate cluster. Inhibits AXI transactions from cluster to SoC: b0: Disable; b1: Enable

PADFUN0

Pad mux config register (pads 0 to 15)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_0	0x1	Selects between: hyper0_ckn / gpio0
3:2	R/W	PADMUX_1	0x1	Selects between: hyper0_ck / gpio1
5:4	R/W	PADMUX_2	0x1	Selects between: hyper0_dq0 / gpio2
7:6	R/W	PADMUX_3	0x1	Selects between: hyper0_dq1 / gpio3
9:8	R/W	PADMUX_4	0x1	Selects between: hyper0_dq2 / gpio4
11:10	R/W	PADMUX_5	0x1	Selects between: hyper0_dq3 / gpio5
13:12	R/W	PADMUX_6	0x1	Selects between: hyper0_dq4 / gpio6
15:14	R/W	PADMUX_7	0x1	Selects between: hyper0_dq5 / gpio7
17:16	R/W	PADMUX_8	0x1	Selects between: hyper0_dq6 / gpio8
19:18	R/W	PADMUX_9	0x1	Selects between: hyper0_dq7 / gpio9
21:20	R/W	PADMUX_10	0x1	Selects between: hyper0_csn0 / gpio10
23:22	R/W	PADMUX_11	0x1	Selects between: hyper0_csn1 / gpio11
25:24	R/W	PADMUX_12	0x1	Selects between: hyper0_rwds / gpio12
27:26	R/W	PADMUX_13	0x1	Selects between: hyper1_ckn / gpio13
29:28	R/W	PADMUX_14	0x1	Selects between: hyper1_ck / gpio14
31:30	R/W	PADMUX_15	0x1	Selects between: hyper1_dq0 / gpio15

PADFUN1

Pad mux config register (pads 16 to 31)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_16	0x1	Selects between: hyper1_dq1 / gpio16
3:2	R/W	PADMUX_17	0x1	Selects between: hyper1_dq2 / gpio17
5:4	R/W	PADMUX_18	0x1	Selects between: hyper1_dq3 / gpio18
7:6	R/W	PADMUX_19	0x1	Selects between: hyper1_dq4 / gpio19
9:8	R/W	PADMUX_20	0x1	Selects between: hyper1_dq5 / gpio20
11:10	R/W	PADMUX_21	0x1	Selects between: hyper1_dq6 / gpio21
13:12	R/W	PADMUX_22	0x1	Selects between: hyper1_dq7 / gpio22
15:14	R/W	PADMUX_23	0x1	Selects between: hyper1_csn0 / gpio23
17:16	R/W	PADMUX_24	0x1	Selects between: hyper1_csn1 / gpio24
19:18	R/W	PADMUX_25	0x1	Selects between: hyper1_rwds / gpio25
21:20	R/W	PADMUX_26	0x1	Selects between: spi0_sck / gpio26
23:22	R/W	PADMUX_27	0x1	Selects between: mux_group_sel_spi0_cs0 / gpio27
25:24	R/W	PADMUX_28	0x1	Selects between: mux_group_sel_spi0_cs1 / gpio28
27:26	R/W	PADMUX_29	0x1	Selects between: mux_group_sel_spi0_cs2 / gpio29
29:28	R/W	PADMUX_30	0x1	Selects between: mux_group_sel_spi0_cs3 / gpio30
31:30	R/W	PADMUX_31	0x1	Selects between: spi0_sdo / gpio31

PADFUN2

Pad mux config register (pads 32 to 47)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_32	0x1	Selects between: spi0_sdi / gpio32
3:2	R/W	PADMUX_33	0x1	Selects between: spi1_sck / gpio33 / uart3_clk
5:4	R/W	PADMUX_34	0x1	Selects between: mux_group_sel_spi1_cs0 / gpio34
7:6	R/W	PADMUX_35	0x1	Selects between: mux_group_sel_spi1_cs1 / gpio35
9:8	R/W	PADMUX_36	0x1	Selects between: spi1_cs2 / gpio36 / uart3_cts / spi1_sdio2
11:10	R/W	PADMUX_37	0x1	Selects between: spi1_cs3 / gpio37 / uart3_rts / spi1_sdio3
13:12	R/W	PADMUX_38	0x1	Selects between: spi1_sdo / gpio38
15:14	R/W	PADMUX_39	0x1	Selects between: spi1_sdi / gpio39
17:16	R/W	PADMUX_40	0x1	Selects between: mux_group_sel_i2c0_sda / gpio40
19:18	R/W	PADMUX_41	0x1	Selects between: mux_group_sel_i2c0_scl / gpio41
21:20	R/W	PADMUX_42	0x1	Selects between: mux_group_sel_i2c1_sda / gpio42
23:22	R/W	PADMUX_43	0x1	Selects between: mux_group_sel_i2c1_scl / gpio43
25:24	R/W	PADMUX_44	0x1	Selects between: mux_group_sel_i2c2_sda / gpio44
27:26	R/W	PADMUX_45	0x1	Selects between: mux_group_sel_i2c2_scl / gpio45
29:28	R/W	PADMUX_46	0x1	Selects between: i3c_sda / gpio46 / i2c3_sda / spi0_sdio2
31:30	R/W	PADMUX_47	0x1	Selects between: i3c_scl / gpio47 / i2c3_scl / spi0_sdio3

PADFUN3

Pad mux config register (pads 48 to 63)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_48	0x1	Selects between: i2s0_sck / gpio48 / uart2_clk
3:2	R/W	PADMUX_49	0x1	Selects between: i2s0_ws / gpio49
5:4	R/W	PADMUX_50	0x1	Selects between: i2s0_sdi / gpio50
7:6	R/W	PADMUX_51	0x1	Selects between: i2s0_sdo / gpio51
9:8	R/W	PADMUX_52	0x1	Selects between: i2s1_sck / gpio52
11:10	R/W	PADMUX_53	0x1	Selects between: i2s1_ws / gpio53 / spi2_cs1
13:12	R/W	PADMUX_54	0x1	Selects between: i2s1_sdi / gpio54 / spi2_cs2
15:14	R/W	PADMUX_55	0x1	Selects between: i2s1_sdo / gpio55 / spi2_cs3
17:16	R/W	PADMUX_56	0x1	Selects between: i2s2_sck / gpio56 / spi2_sck
19:18	R/W	PADMUX_57	0x1	Selects between: i2s2_ws / gpio57 / spi2_cs0
21:20	R/W	PADMUX_58	0x1	Selects between: i2s2_sdi / gpio58 / spi2_sdi
23:22	R/W	PADMUX_59	0x1	Selects between: i2s2_sdo / gpio59 / spi2_sdo
25:24	R/W	PADMUX_60	0x1	Selects between: mux_group_sel_uart0_rx / gpio60
27:26	R/W	PADMUX_61	0x1	Selects between: mux_group_sel_uart0_tx / gpio61
29:28	R/W	PADMUX_62	0x1	Selects between: mux_group_sel_uart0_cts / gpio62
31:30	R/W	PADMUX_63	0x1	Selects between: mux_group_sel_uart0_rts / gpio63

PADFUN4

Pad mux config register (pads 64 to 79)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_64	0x1	Selects between: uart0_clk / gpio64
3:2	R/W	PADMUX_65	0x1	Selects between: mux_group_sel_uart1_rx / gpio65
5:4	R/W	PADMUX_66	0x1	Selects between: mux_group_sel_uart1_tx / gpio66
7:6	R/W	PADMUX_67	0x1	Selects between: mux_group_sel_pwm0 / gpio67
9:8	R/W	PADMUX_68	0x1	Selects between: mux_group_sel_pwm1 / gpio68
11:10	R/W	PADMUX_69	0x1	Selects between: uart1_clk / gpio69
13:12	R/W	PADMUX_70	0x1	Selects between: cam_pclk / gpio70 / spi3_sck
15:14	R/W	PADMUX_71	0x1	Selects between: cam_hsync / gpio71 / spi3_cs0 / csi2_hsync
17:16	R/W	PADMUX_72	0x1	Selects between: cam_data0 / gpio72 / spi3_cs1
19:18	R/W	PADMUX_73	0x1	Selects between: cam_data1 / gpio73 / spi3_cs2
21:20	R/W	PADMUX_74	0x1	Selects between: cam_data2 / gpio74 / spi3_cs3
23:22	R/W	PADMUX_75	0x1	Selects between: cam_data3 / gpio75 / spi3_sdo
25:24	R/W	PADMUX_76	0x1	Selects between: cam_data4 / gpio76 / spi3_sdi
27:26	R/W	PADMUX_77	0x1	Selects between: cam_data5 / gpio77 / observability1
29:28	R/W	PADMUX_78	0x1	Selects between: cam_data6 / gpio78 / observability2
31:30	R/W	PADMUX_79	0x1	Selects between: cam_data7 / gpio79 / observability3

PADFUN5

Pad mux config register (pads 80 to 89)

Bit #	R/W	Name	Reset	Description
1:0	R/W	PADMUX_80	0x1	Selects between: cam_vsync / gpio80 / observability4 / csi2_vsync
3:2	R/W	PADMUX_81	0x0	Selects between: jtag_tck / gpio81 / uart4_clk
5:4	R/W	PADMUX_82	0x0	Selects between: jtag_tdi / gpio82 / uart4_rx
7:6	R/W	PADMUX_83	0x0	Selects between: jtag_tdo / gpio83 / uart4_tx
9:8	R/W	PADMUX_84	0x0	Selects between: jtag_tms / gpio84 / uart4_cts
11:10	R/W	PADMUX_85	0x0	Selects between: jtag_trst / gpio85 / uart4_rts
13:12	R/W	PADMUX_86	0x1	Selects between: wakeup_spi2_sck / gpio86
15:14	R/W	PADMUX_87	0x1	Selects between: wakeup_spi2_sdi / gpio87
17:16	R/W	PADMUX_88	0x1	Selects between: wakeup_spi2_sdo / gpio88
19:18	R/W	PADMUX_89	0x1	Selects between: wakeup_spi2_cs0 / gpio89

FEATURE_DISABLE_QK

Feature disablement for Quiddikey features

Bit #	R/W	Name	Reset	Description
0	R/W	DISABLE_QUIDDIKEY_UNWRAP	0x0	Disable Quiddikey unwrap
1	R/W	DISABLE_QUIDDIKEY_ENROLL	0x0	Disable Quiddikey enroll
31	R/W	DISABLE_LOCK	0x0	When set, DISABLE_* registers cannot be written to zero. Configuration is lost when SoC domain is switched off

PADCFG0

Pad function register (pads 0 to 3)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_0_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_0_DRIVE_STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_1_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_1_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_1_DRIVE_STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_2_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_2_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_2_DRIVE_STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_3_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_3_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_3_DRIVE_STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG1

Pad function register (pads 4 to 7)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_4_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_4_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_4_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_5_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_5_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_5_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_6_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_6_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_6_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_7_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_7_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_7_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG2

Pad function register (pads 8 to 11)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_8_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_8_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_8_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_9_PULL_DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_9_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_9_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_10_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_10_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_10_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_11_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_11_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_11_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG3

Pad function register (pads 12 to 15)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_12_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_12_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_12_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_13_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_13_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_13_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_14_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_14_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_14_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_15_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_15_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_15_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG4

Pad function register (pads 16 to 19)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_16_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_16_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_16_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_17_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_17_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_17_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_18_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_18_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_18_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_19_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_19_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_19_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG5

Pad function register (pads 20 to 23)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_20_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_20_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_20_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_21_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_21_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_21_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_22_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_22_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_22_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
24	R/W	PAD_23_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_23_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_23_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA

PADCFG6

Pad function register (pads 24 to 27)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_24_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_24_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_24_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
8	R/W	PAD_25_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_25_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_25_DRIVE_-STRENGTH	0x0	0: 1mA, 1: 2mA, 2: 4mA, 3: 8mA
16	R/W	PAD_26_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_26_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_26_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_26_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_26_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_27_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_27_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_27_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_27_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_27_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG7

Pad function register (pads 28 to 31)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_28_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_28_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_28_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_28_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_28_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_29_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_29_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_29_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_29_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_29_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_30_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_30_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_30_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_30_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_30_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_31_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_31_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_31_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_31_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_31_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG8

Pad function register (pads 32 to 35)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_32_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_32_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_32_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_32_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_32_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_33_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_33_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_33_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_33_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_33_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_34_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_34_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_34_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_34_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_34_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_35_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_35_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_35_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_35_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_35_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG9

Pad function register (pads 36 to 39)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_36_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_36_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_36_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_36_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_36_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_37_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_37_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_37_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_37_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_37_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_38_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_38_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_38_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_38_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_38_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_39_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_39_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_39_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_39_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_39_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG10

Pad function register (pads 40 to 43)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_40_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_40_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_40_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_40_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_40_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_41_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_41_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_41_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_41_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_41_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_42_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_42_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_42_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_42_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_42_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_43_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_43_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_43_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_43_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_43_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG11

Pad function register (pads 44 to 47)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_44_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_44_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_44_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_44_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_44_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_45_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_45_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_45_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_45_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_45_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_46_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_46_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_46_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_46_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_46_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_47_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_47_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_47_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_47_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_47_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG12

Pad function register (pads 48 to 51)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_48_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_48_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_48_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_48_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_48_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_49_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_49_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_49_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_49_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_49_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_50_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_50_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_50_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_50_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_50_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_51_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_51_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_51_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_51_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_51_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG13

Pad function register (pads 52 to 55)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_52_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_52_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_52_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_52_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_52_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_53_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_53_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_53_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_53_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_53_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_54_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_54_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_54_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_54_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_54_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_55_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_55_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_55_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_55_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_55_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG14

Pad function register (pads 56 to 59)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_56_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_56_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_56_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_56_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_56_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_57_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_57_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_57_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_57_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_57_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_58_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_58_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_58_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_58_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_58_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_59_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_59_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_59_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_59_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_59_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG15

Pad function register (pads 60 to 63)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_60_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_60_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_60_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_60_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_60_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_61_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_61_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_61_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_61_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_61_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_62_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_62_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_62_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_62_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_62_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_63_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_63_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_63_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_63_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_63_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG16

Pad function register (pads 64 to 67)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_64_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_64_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_64_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_64_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_64_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_65_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_65_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_65_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_65_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_65_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_66_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_66_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_66_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_66_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_66_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_67_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_67_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_67_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_67_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_67_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG17

Pad function register (pads 68 to 71)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_68_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_68_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_68_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_68_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_68_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_69_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_69_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_69_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_69_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_69_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_70_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_70_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_70_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_70_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_70_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_71_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_71_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_71_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_71_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_71_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG18

Pad function register (pads 72 to 75)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_72_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_72_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_72_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_72_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_72_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_73_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_73_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_73_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_73_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_73_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_74_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_74_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_74_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_74_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_74_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_75_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_75_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_75_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_75_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_75_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG19

Pad function register (pads 76 to 79)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_76_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_76_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_76_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_76_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_76_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_77_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_77_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_77_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_77_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_77_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_78_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_78_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_78_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_78_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_78_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_79_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_79_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_79_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_79_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_79_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG20

Pad function register (pads 80 to 83)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_80_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_80_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_80_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_80_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_80_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_81_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_81_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_81_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_81_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_81_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_82_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_82_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_82_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_82_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_82_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_83_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_83_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_83_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_83_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_83_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG21

Pad function register (pads 84 to 87)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_84_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_84_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_84_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_84_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_84_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_85_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_85_PULL_UP	0x1	Write 1 to enable I/O pull-up
11:10	R/W	PAD_85_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_85_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_85_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
16	R/W	PAD_86_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
17	R/W	PAD_86_PULL_UP	0x0	Write 1 to enable I/O pull-up
19:18	R/W	PAD_86_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
20	R/W	PAD_86_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
21	R/W	PAD_86_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
24	R/W	PAD_87_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
25	R/W	PAD_87_PULL_UP	0x0	Write 1 to enable I/O pull-up
27:26	R/W	PAD_87_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
28	R/W	PAD_87_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
29	R/W	PAD_87_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

PADCFG22

Pad function register (pads 88 to 89)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_88_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
1	R/W	PAD_88_PULL_UP	0x0	Write 1 to enable I/O pull-up
3:2	R/W	PAD_88_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
4	R/W	PAD_88_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
5	R/W	PAD_88_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)
8	R/W	PAD_89_PULL_-DOWN	0x0	Write 1 to enable I/O pull-down
9	R/W	PAD_89_PULL_UP	0x0	Write 1 to enable I/O pull-up
11:10	R/W	PAD_89_DRIVE_-STRENGTH	0x0	0: 2mA, 1: 4mA, 2: 8mA, 3: 12mA
12	R/W	PAD_89_SCHMITT_-TRIGGER	0x0	Write 1 to enable Schmitt trigger on this I/O
13	R/W	PAD_89_SLEW_RATE	0x0	0: When VDDIO = 1.8V, 1: When VDDIO = 1.5/1.2V (unused)

REG_REPROG_PAD0

Controls reprogrammable pads 27, 28, 29, 30 and 34

Bit #	R/W	Name	Reset	Description
5:0	R/W	MUX_GROUP_SEL_-SPI0_CS0	0x00	Selects function for reprogrammable pad 27. Default function: SPI0_CS0
11:6	R/W	MUX_GROUP_SEL_-SPI0_CS1	0x01	Selects function for reprogrammable pad 28. Default function: SPI0_CS1
17:12	R/W	MUX_GROUP_SEL_-SPI0_CS2	0x02	Selects function for reprogrammable pad 29. Default function: SPI0_CS2
23:18	R/W	MUX_GROUP_SEL_-SPI0_CS3	0x03	Selects function for reprogrammable pad 30. Default function: SPI0_CS3
29:24	R/W	MUX_GROUP_SEL_-SPI1_CS0	0x04	Selects function for reprogrammable pad 34. Default function: SPI1_CS0

REG_REPROG_PAD1

Controls reprogrammable pads 35, 40, 41, 42 and 43

Bit #	R/W	Name	Reset	Description
5:0	R/W	MUX_GROUP_SEL_-SPI1_CS1	0x06	Selects function for reprogrammable pad 35. Default function: SPI1_CS1
11:6	R/W	MUX_GROUP_SEL_-I2C0_SDA	0x08	Selects function for reprogrammable pad 40. Default function: I2C0_SDA
17:12	R/W	MUX_GROUP_SEL_-I2C0_SCL	0x09	Selects function for reprogrammable pad 41. Default function: I2C0_SCL
23:18	R/W	MUX_GROUP_SEL_-I2C1_SDA	0x0A	Selects function for reprogrammable pad 42. Default function: I2C1_SDA
29:24	R/W	MUX_GROUP_SEL_-I2C1_SCL	0x0C	Selects function for reprogrammable pad 43. Default function: I2C1_SCL

REG_REPROG_PAD2

Controls reprogrammable pads 44, 45, 60, 61 and 62

Bit #	R/W	Name	Reset	Description
5:0	R/W	MUX_GROUP_SEL_-I2C2_SDA	0x0E	Selects function for reprogrammable pad 44. Default function: I2C2_SDA
11:6	R/W	MUX_GROUP_SEL_-I2C2_SCL	0x10	Selects function for reprogrammable pad 45. Default function: I2C2_SCL
17:12	R/W	MUX_GROUP_SEL_-UART0_RX	0x12	Selects function for reprogrammable pad 60. Default function: UART0_RX
23:18	R/W	MUX_GROUP_SEL_-UART0_TX	0x13	Selects function for reprogrammable pad 61. Default function: UART0_TX
29:24	R/W	MUX_GROUP_SEL_-UART0_CTS	0x14	Selects function for reprogrammable pad 62. Default function: UART0_CTS

REG_REPROG_PAD3

Controls reprogrammable pads 63, 65, 66, 67 and 68

Bit #	R/W	Name	Reset	Description
5:0	R/W	MUX_GROUP_SEL_-UART0_RTS	0x16	Selects function for reprogrammable pad 63. Default function: UART0_RTS
11:6	R/W	MUX_GROUP_SEL_-UART1_RX	0x18	Selects function for reprogrammable pad 65. Default function: UART1_RX
17:12	R/W	MUX_GROUP_SEL_-UART1_TX	0x1B	Selects function for reprogrammable pad 66. Default function: UART1_TX
23:18	R/W	MUX_GROUP_SEL_-PWM0	0x1E	Selects function for reprogrammable pad 67. Default function: PWM0
29:24	R/W	MUX_GROUP_SEL_-PWM1	0x21	Selects function for reprogrammable pad 68. Default function: PWM1

CL_BUSY

Cluster busy register

Bit #	R/W	Name	Reset	Description
0	R	BUSY	0x0	Cluster busy flag (i.e. It's 1 if there is at least 1 active block in the cluster)

JTAGREG

JTAG external register

Bit #	R/W	Name	Reset	Description
3:0	R/W	INTERNAL	0x0	JTAG internal register used for synchronisation from external debugger
11:8	R	EXTERNAL	0x0	JTAG external register used for synchronisation from external debugger

REF_FAST_CLK_DIV

Read only, reference fast clk divided by power of 2

Bit #	R/W	Name	Reset	Description
0	R	CLK	0x0	Current value of the divided REF FAST clock

SW_RST

Software reset, reboot

Bit #	R/W	Name	Reset	Description
0	R	RESET	0x0	Writing 1 triggers a chip reset

CORESTATUS

EOC and chip status register

Bit #	R/W	Name	Reset	Description
31:0	R/W	STATUS	0x0	Chip status register. The SW can store the exit value value so that the external loader can get it
31	R/W	EOC	0x0	End Of Computation. The SW can store 1 here to notify the external loader that the execution is finished

BOOTSEL

Value of bootsel pads

Bit #	R/W	Name	Reset	Description
1:0	R	BOOTSEL	0x0	Value of the bootsel pads. It is used by the ROM software to select the boot mode (see GAP9 ROM documentation for more details)

WD_RST_RST

Rearm WD timeout

Bit #	R/W	Name	Reset	Description
31:0	W	CLEAR	0x0	Any write to this register re-arms the watchdog timeout counter

WD_RST_SET

Set WD timer

Bit #	R/W	Name	Reset	Description
23:0	R/W	REF_COUNTER	0xFFFFFFF	Watchdog timeout configuration, in periods of the SOC slow clock. The maximum duration when using REF SLOW clock is $\frac{2^{24}-1}{32.768kHz}$ = 8min32s. Note: when read, returns the timeout configuration, not current counter value
31	R/W	ENABLE	0x0	Enable the watchdog: triggers chip reset if the timeout counter expires without being re-armed

RWM_CSI2

Margin adjust settings for CSI-2 Controller

Bit #	R/W	Name	Reset	Description
5:0	R/W	RWM_CSI2	0x00	Margin settings for CSI-2 controller memories

IDLE_MODE

Activates IDLE MODE

Bit #	R/W	Name	Reset	Description
15:0	R/W	REF_COUNTER	0xFFFF	IDLE mode duration, in periods of the SOC slow clock. The maximum duration when using REF SLOW clock is $\frac{2^{16}-1}{32.768kHz}$ = 2s
31	R/W	ENABLE	0x0	Write 1 to enable IDLE mode. Read returns 0 when IDLE mode has ended

RWM_ANC

Margin adjust settings for SFU

Bit #	R/W	Name	Reset	Description
5:0	R/W	RWM_ASRC_DPRAM	0x00	Margin settings for SFU ASRC dual-port memories
12:6	R/W	RWM_ASRC_RAM	0x40	Margin settings for SFU ASRC single-port memories
18:13	R/W	RWM_GFU_DPRAM	0x00	Margin settings for SFU GFU dual-port memories
25:19	R/W	RWM_POLY_RAM	0x40	Margin settings for SFU Polyphase Filter memories
29:26	R/W	RWM_GFU_R1PL	0x00	Margin settings for SFU GFU single-port memories

REF_CLK_MUX

Reference clock selection

Bit #	R/W	Name	Reset	Description
0	R/W	SOC	0x0	Reference clock selection for SOC: 0: 32kHz REF SLOW clock; 1: divided REF FAST clock
1	R/W	CLUSTER	0x0	Reference clock selection for cluster: 0: 32kHz REF SLOW clock; 1: divided REF FAST clock

SUPERVISOR_DBG

Bit #	R/W	Name	Reset	Description
0	R/W	MODE	0x0	Write 1 to enable supervisor debug mode

DBG_CTRL

Debug access control

Bit #	R/W	Name	Reset	Description
0	R/W	HART_CL_CORE0	0x0	Write 1 to enable HART for cluster core 0
1	R/W	HART_CL_CORE1	0x0	Write 1 to enable HART for cluster core 1
2	R/W	HART_CL_CORE2	0x0	Write 1 to enable HART for cluster core 2
3	R/W	HART_CL_CORE3	0x0	Write 1 to enable HART for cluster core 3
4	R/W	HART_CL_CORE4	0x0	Write 1 to enable HART for cluster core 4
5	R/W	HART_CL_CORE5	0x0	Write 1 to enable HART for cluster core 5
6	R/W	HART_CL_CORE6	0x0	Write 1 to enable HART for cluster core 6
7	R/W	HART_CL_CORE7	0x0	Write 1 to enable HART for cluster core 7
8	R/W	HART_CL_CORE8	0x0	Write 1 to enable HART for cluster core 8
9	R/W	HART_FC_CORE	0x1	Write 1 to enable HART for FC core

CLK_DIV_I3C

Clock divider for I3C

Bit #	R/W	Name	Reset	Description
0	R/W	CLK_EN	0x0	Write 1 to enable clock for I3C
15:8	R/W	CLK_DIVIDER	0x0	Integer divider for I3C

CLK_EN_QUIDDIKEY

Quiddikey enablement

Bit #	R/W	Name	Reset	Description
0	R/W	QK_CLK_EN	0x0	Write 1 to enable clock for the Quiddikey
1	R/W	QK_RSTN	0x0	Control the value of the active low reset of Quiddikey

SLEEP_CTRL_INFO

Read-only access to sleep status

Bit #	R/W	Name	Reset	Description
7:0	R	REBOOT	0x0	SW config. This field is only interpreted by ROM and runtime to keep information accross deep sleep
8	R	RTC_WAKE_EN	0x0	Enable RTC wakeup
9	R	RTC_EVENT	0x0	RTC event
10	R	EXT_WAKEUP_EN	0x0	Enable external wakeup
26:11	R	EXT_EVENT	0x0	External event on wake-up I/Os
28:27	R	WAKEUP_CFG	0x0	Selected wake-up sequence
29	R	FORCE_AO_PADS	0x0	Force always-on I/Os into their sleep mode configuration (see SLEEP_PAD_CFG* registers)
30	R	CNT_WAKE_EN	0x0	Enable counter wakeup
31	R	CNT_EVENT	0x0	Counter event

VERSION

Show chip version (User controlled)

Bit #	R/W	Name	Reset	Description
3:0	R/W	VERSION	0x0	4-bit field available to SW to store chip version

SLEEP_SPIS_CTRL

(*Always-on register*) Sleep SPIS control

Bit #	R/W	Name	Reset	Description
0	R/W	MUX	0x0	Enable wake-up interface: 0: SPIS wake-up enabled; 1: GPIO-only external wake-up
2:1	R/W	SPIS_MODE	0x0	SPI slave controller CPOL and CPHA
3	R/W	SPIS_RSTN	0x1	Reset the SPI Slave wakeup controller
4	R/W	SPIS_SW_DONE	0x0	Notify outside device wakeup done

SLEEP_CTRL

(*Always-on register*) Sleep control

Bit #	R/W	Name	Reset	Description
7:0	R/W	REBOOT	0x0	SW config. This field is only interpreted by ROM and runtime to keep information accross deep sleep
8	R/W	RTC_WAKE_EN	0x0	Enable RTC wakeup
9	R	RTC_EVENT	0x0	RTC event
10	R/W	EXT_WAKEUP_EN	0x0	Enable external wakeup
26:11	R	EXT_EVENT	0x0	External event. Bit order corresponds to the numbering in SLEEP_GPIO_CTRL register
28:27	R/W	WAKEUP_CFG	0x0	Selected wake-up sequence
29	R/W	FORCE_AO_PADS	0x0	Force always-on I/Os into their sleep mode configuration (see SLEEP_PAD_CFG* registers)

SLEEP_GPIO_CTRL

(*Always-on register*) Sleep GPIO control

Bit #	R/W	Name	Reset	Description
1:0	R/W	GPIO_WAKEUP_0_-TYPE	0x0	Wake-up on I2C1_SDA pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
3:2	R/W	GPIO_WAKEUP_1_-TYPE	0x0	Wake-up on I2C1_SCL pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
5:4	R/W	GPIO_WAKEUP_2_-TYPE	0x0	Wake-up on UART1_RX pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
7:6	R/W	GPIO_WAKEUP_3_-TYPE	0x0	Wake-up on UART1_TX pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
9:8	R/W	GPIO_WAKEUP_4_-TYPE	0x0	Wake-up on PWM0 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
11:10	R/W	GPIO_WAKEUP_5_-TYPE	0x0	Wake-up on PWM1 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
13:12	R/W	GPIO_WAKEUP_6_-TYPE	0x0	Wake-up on WAKEUP_CS0 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
15:14	R/W	GPIO_WAKEUP_7_-TYPE	0x0	Wake-up on WAKEUP_SDO pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
17:16	R/W	GPIO_WAKEUP_8_-TYPE	0x0	Wake-up on I2C0_SDA pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
19:18	R/W	GPIO_WAKEUP_9_-TYPE	0x0	Wake-up on I2C0_SCL pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
21:20	R/W	GPIO_WAKEUP_10_-TYPE	0x0	Wake-up on SPI1_CS0 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
23:22	R/W	GPIO_WAKEUP_11_-TYPE	0x0	Wake-up on SPI1_CS1 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
25:24	R/W	GPIO_WAKEUP_12_-TYPE	0x0	Wake-up on SPI1_CS2 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
27:26	R/W	GPIO_WAKEUP_13_-TYPE	0x0	Wake-up on SPI1_CS3 pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
29:28	R/W	GPIO_WAKEUP_14_-TYPE	0x0	Wake-up on SPI1_SDI pad. b00: disabled; b01: wake on high signal; b10: wake on low signal
31:30	R/W	GPIO_WAKEUP_15_-TYPE	0x0	Wake-up on SPI1_SDO pad. b00: disabled; b01: wake on high signal; b10: wake on low signal

SLEEP_CNT_CTRL

(*Always-on register*) Sleep counter control

Bit #	R/W	Name	Reset	Description
19:0	R/W	VALUE	0x0	Counter target value
20	R/W	EN	0x0	Counter enable
21	R/W	CNT_EVENT	0x0	Counter event (target value reached)

REG_OSC_CTRL

(*Always-on register*) Controls fast oscillator

Bit #	R/W	Name	Reset	Description
0	R/W	FAST_OSC_EN	0x0	0: Power Down, 1: Enables fast oscillator
1	R/W	SLOW_OSC_EN	0x0	0: Power Down, 1: Enables 32kHz oscillator
2	R/W	BYPASS_FAST_OSC	0x0	0: Use fast oscillator to generate FAST REF clock, 1: Use external signal on USART0_CLK/FAST_REF_CK pad as FAST REF clock

CLK_DIV_REF_FAST_POW2

(*Always-on register*) Controls fast oscillator pow2 divider

Bit #	R/W	Name	Reset	Description
2:0	R/W	DIVIDER	0x0	Fast clock divider (division is 2^DIVIDER)
3	R/W	EN	0x0	Enable divider

FEATURE_DISABLE

(*Always-on register*) Feature disablement from always on (safe) domain

Bit #	R/W	Name	Reset	Description
0	R/W	DISABLE_JTAG	0x1	Disable JTAG. It is disabled at reset and may be enabled back by the bootloader
1	R/W	DISABLE_CRYPTO	0x0	Disable AES and Quiddikey
2	R/W	DISABLE_MRAM	0x0	Disable mram
3	R/W	DISABLE_EFUSE_PROGRAM	0x0	Disable eFuse programming
4	R/W	DISABLE_SAFE_JTAG	0x0	Disable safe domain JTAG to avoid accidental toggling when alternates are used
31	R/W	DISABLE_LOCK	0x0	When set, DISABLE_* registers cannot be written to zero until next whole chip reset or power-up. Configuration is kept in sleep modes

SLEEP_PAD_CFG0

(*Always-on register*) Pad control during sleep (pads 33 to 36)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 33 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 33 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 33 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 33 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 33 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 34 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 34 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 34 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 34 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 34 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 35 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 35 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 35 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 35 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 35 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 36 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 36 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 36 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 36 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 36 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG1

(*Always-on register*) Pad control during sleep (pads 37 to 40)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 37 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 37 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 37 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 37 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 37 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 38 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 38 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 38 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 38 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 38 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 39 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 39 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 39 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 39 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 39 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 40 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 40 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 40 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 40 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 40 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG2

(*Always-on register*) Pad control during sleep (pads 41 to 43 and 65)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 41 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 41 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 41 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 41 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 41 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 42 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 42 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 42 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 42 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 42 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 43 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 43 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 43 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 43 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 43 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 65 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 65 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 65 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 65 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 65 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG3

(*Always-on register*) Pad control during sleep (pads 66 to 69)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 66 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 66 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 66 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 66 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 66 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 67 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 67 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 67 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 67 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 67 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 68 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 68 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 68 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 68 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 68 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 69 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 69 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 69 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 69 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 69 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG4

(*Always-on register*) Pad control during sleep (pads 81 to 84)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 81 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 81 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 81 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 81 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 81 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 82 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 82 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 82 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 82 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 82 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 83 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 83 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 83 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 83 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 83 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 84 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 84 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 84 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 84 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 84 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG5

(*Always-on register*) Pad control during sleep (pads 85 to 88)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 85 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x1	Pad 85 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 85 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 85 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 85 output enabled in deep or retentive sleep mode
8	R/W	PAD_1_PULL_DOWN	0x0	Pad 86 pull-down enabled in deep or retentive sleep mode
9	R/W	PAD_1_PULL_UP	0x0	Pad 86 pull-up enabled in deep or retentive sleep mode
11:10	R/W	PAD_1_DRIVE	0x0	Pad 86 driver strength in deep or retentive sleep mode
12	R/W	PAD_1_OUT	0x0	Pad 86 output value when output is enabled in deep or retentive sleep mode
13	R/W	PAD_1_OEN	0x1	Pad 86 output enabled in deep or retentive sleep mode
16	R/W	PAD_2_PULL_DOWN	0x0	Pad 87 pull-down enabled in deep or retentive sleep mode
17	R/W	PAD_2_PULL_UP	0x0	Pad 87 pull-up enabled in deep or retentive sleep mode
19:18	R/W	PAD_2_DRIVE	0x0	Pad 87 driver strength in deep or retentive sleep mode
20	R/W	PAD_2_OUT	0x0	Pad 87 output value when output is enabled in deep or retentive sleep mode
21	R/W	PAD_2_OEN	0x1	Pad 87 output enabled in deep or retentive sleep mode
24	R/W	PAD_3_PULL_DOWN	0x0	Pad 88 pull-down enabled in deep or retentive sleep mode
25	R/W	PAD_3_PULL_UP	0x0	Pad 88 pull-up enabled in deep or retentive sleep mode
27:26	R/W	PAD_3_DRIVE	0x0	Pad 88 driver strength in deep or retentive sleep mode
28	R/W	PAD_3_OUT	0x0	Pad 88 output value when output is enabled in deep or retentive sleep mode
29	R/W	PAD_3_OEN	0x1	Pad 88 output enabled in deep or retentive sleep mode

SLEEP_PAD_CFG6

(*Always-on register*) Pad control during sleep (pad 89)

Bit #	R/W	Name	Reset	Description
0	R/W	PAD_0_PULL_DOWN	0x0	Pad 89 pull-down enabled in deep or retentive sleep mode
1	R/W	PAD_0_PULL_UP	0x0	Pad 89 pull-up enabled in deep or retentive sleep mode
3:2	R/W	PAD_0_DRIVE	0x0	Pad 89 driver strength in deep or retentive sleep mode
4	R/W	PAD_0_OUT	0x0	Pad 89 output value when output is enabled in deep or retentive sleep mode
5	R/W	PAD_0_OEN	0x1	Pad 89 output enabled in deep or retentive sleep mode

L2_CTRL_ACTIVE

(*Always-on register*) Controls L2 power when SOC is powered on

Bit #	R/W	Name	Reset	Description
11:0	R/W	INTLVD_L2_POWER-GATE	0x0	Value of the POWERGATE pin of L2 memory cuts of INTERLEAVED RAM banks when SOC domain is on (should always be 0)
27:16	R/W	INTLVD_L2_DEEP-SLEEP	0x0	Value of the DEEPSLEEP pin of L2 memory cuts of INTERLEAVED RAM banks when SOC domain is on (set to 1 to force deepsleep mode – data are lost)

L2_PWR_ACTIVE

(*Always-on register*) Controls L2 power when SOC is powered on

Bit #	R/W	Name	Reset	Description
11:0	R/W	INTLVD_L2_SD_AR-RAY	0x0	Power switch control for memory array of INTER-LEAVED banks when SOC domain is on: bit i=1 shuts down array supply for bank i
27:16	R/W	INTLVD_L2_SD_PE-RIPH	0x0	Power switch control for memory periphery of IN-TERLEAVED banks when SOC domain is on: bit i=1 shuts down periphery supply for bank i

NEVACFG

(*Always-on register*) NEVA config

Bit #	R/W	Name	Reset	Description
4:0	R/W	NEVA_IO_LS_TSEL	0x1F	Max output current when switching on switchable I/Os
5	R/W	NEVA_IO_LS_FORCE	0x0	Force NEVA switch status for switchable I/Os
6	R/W	NEVA_IO_LS_EN	0x0	EN signal value for switchable I/Os in force mode
7	R/W	NEVA_IO_LS_SD	0x0	SD signal value for switchable I/Os in force mode
12:8	R/W	NEVA_IO_HS_TSEL	0x1F	Max output current when switching on memory interfaces
13	R/W	NEVA_IO_HS_FORCE	0x0	Force NEVA switch status for memory interfaces
14	R/W	NEVA_IO_HS_EN	0x0	EN signal value for memory interfaces in force mode
15	R/W	NEVA_IO_HS_SD	0x0	SD signal value for memory interfaces in force mode

TRCCFG

(*Always-on register*) TRC config

Bit #	R/W	Name	Reset	Description
1:0	R/W	CSI2_PROGDELAY	0x0	CSI-2 domain power switch controller PROGDELAY setting
4:2	R/W	CSI2_CURRSET	0x7	CSI-2 domain power switch controller CURRSET setting
6:5	R/W	MRAM_PROGDELAY	0x0	MRAM domain power switch controller PROGDELAY setting
9:7	R/W	MRAM_CURRSET	0x7	MRAM domain power switch controller CURRSET setting
11:10	R/W	SOC_PROGDELAY	0x0	SOC domain power switch controller PROGDELAY setting
14:12	R/W	SOC_CURRSET	0x7	SOC domain power switch controller CURRSET setting
16:15	R/W	CLUSTER_PROGDELAY	0x0	CLUSTER domain power switch controller PROGDELAY setting
19:17	R/W	CLUSTER_CURRSET	0x7	CLUSTER domain power switch controller CURRSET setting
21:20	R/W	SFU_PROGDELAY	0x0	SFU domain power switch controller PROGDELAY setting
23:22	R/W	SFU_CURRSET	0x3	SFU domain power switch controller CURRSET setting (used 3-bit CurrSet value is bit0,bit1,bit0 of this field)
24	R/W	SOC_ABB_DISABLE	0x1	Disable SOC/SFU domains ABB (default at boot is ABB disabled). *For first engineering samples, see ABBCFG register instead.*

RWM_L2_MEM

(*Always-on register*) Read/write margins for L2 and ROM memories

Bit #	R/W	Name	Reset	Description
5:0	R/W	RWM_L2_INTLVD	0x20	Margin setting for L2 interleaved banks
13:8	R/W	RWM_L2_PRIVATE	0x20	Margin setting for L2 private banks
16	R/W	RWM_L2_FORCE	0x0	0: use default RWM values, 1: use RWM_L2_* values
21:20	R/W	RWM_ROM_SAWL	0x2	SAWL margin setting for ROM
22	R/W	RWM_ROM_WL	0x0	WL margin setting for ROM
24	R/W	RWM_ROM_FORCE	0x1	0: use default RWM values, 1: use RWM_ROM_* values

CLU_SW_RSTN

(*Always-on register*) Cluster software reset

Bit #	R/W	Name	Reset	Description
0	R/W	CLUSTER_RSTN	0x1	Cluster software reset (active low)

L2_PWR

(*Always-on register*) Controls L2 power when SOC is off (deep/retentive sleep)

Bit #	R/W	Name	Reset	Description
11:0	R/W	INTLVD_L2_SD_ARRAY	0x0	Power switch control for memory array of INTERLEAVED banks when chip is in retentive mode: bit i=0 shuts down array supply for bank i

L2_CTRL

(*Always-on register*) Controls L2 power when SOC is off (deep/retentive sleep)

Bit #	R/W	Name	Reset	Description
11:0	R/W	INTLVD_L2_DEEPSLEEP	0x0	Force INTERLEAVED banks into deepsleep (data are lost) when chip is in retentive mode: bit i=0 enables deepsleep for bank i
13:12	R/W	PRI_L2_DEEPSLEEP	0x0	Force PRIVATE banks into deepsleep and switch off array supply (data are lost) when chip is in retentive mode: bit 12+i=0 enables deepsleep for bank i

RARMODE

(*Always-on register*) Controls configuration of the DC-DC modulation at low loads

Bit #	R/W	Name	Reset	Description
0	R/W	ACTIVESKIPB	0x0	Enables standard DC-DC behavior (pulse-skipping off, decimation on)
1	R/W	EN_DECIM_SKIP	0x0	Enables decimation when in pulse-skipping mode
2	R/W	DISABLE_VREF	0x0	Disable generation of 0.6V Vref to save power if eM-RAM is not used

ABBCFG

(*Always-on register*) Used to disable adaptive body-bias

Bit #	R/W	Name	Reset	Description
0	R/W	SOC_ABB_DISABLE	0x0	*Only for first engineering samples – see TRCCFG register for production samples.* Disable SOC/SFU domains ABB
1	R/W	CLUSTER_ABB_DIS-ABLE	0x0	Disable CLUSTER domain ABB

L2_ACK

(*Always-on register*) Acknowledge/status signals from L2 memories

Bit #	R/W	Name	Reset	Description
11:0	R	INTLVD_ARRAY_-WAKE_ACK	0x0	Wake status of the arrays of interleaved L2 banks
27:16	R	INTLVD_PERIPH_-WAKE_ACK	0x0	Wake status of the periphery logic of interleaved L2 banks
31:30	R	PRIVATE_ARRAY_-WAKE_ACK	0x0	Wake status of the arrays of private L2 banks

5.2.4.2 Note on Configuring Reprogrammable Pads

The REG_REPROG_PAD* registers are used to select the functionality mapped onto each pad having a reprogrammable function (also see the Pin Description section).

The table below lists the configuration value for each available function. If 2 or more pads are configured for the same function, only the first pad (according to their numbering) is granted this function.

Function	Encoding
SPI0_CS0	0x00
SPI0_CS1	0x01
SPI0_CS2	0x02
SPI0_CS3	0x03
SPI1_CS0	0x04
UART3_TX	0x05
SPI1_CS1	0x06
UART3_RX	0x07
I2C0_SDA	0x08

Function	Encoding
I2C0_SCL	0x09
I2C1_SDA	0x0A
CSI2_CCI_SDA	0x0B
I2C1_SCL	0x0C
CSI2_CCI_SCL	0x0D
I2C2_SDA	0x0E
UART2_CTS	0x0F
I2C2_SCL	0x10
UART2_RTS	0x11
UART0_RX	0x12
UART0_TX	0x13
UART0_CTS	0x14
UART2_RX	0x15
UART0_RTS	0x16
UART2_TX	0x17
UART1_RX	0x18
PWM2	0x19
PWM4	0x1A
UART1_TX	0x1B
PWM3	0x1C
PWM5	0x1D
PWM0	0x1E
UART1_CTS	0x1F
PWM6	0x20
PWM1	0x21
UART1_RTS	0x22
PWM7	0x23

Note: Due to limitations in GAP9 hardware, when a function, mapped by default to reprogrammable I/O *N*, is remapped to reprogrammable I/O *M*, one must ensure that I/O *N* is still used in alternate 0 configuration (i.e. not used in GPIO mode), else the input from I/O *M* will not be taken into account.

We recommend the following strategy to program functions on the reprogrammable I/Os, which guarantees a working configuration:

1. Identify all the remappable functions needed in the system.
2. First select I/Os where required functions are already mapped.
3. For unmapped functions, select unused reprogrammable I/Os to be used.
4. If for system integration reasons a remapping is desirable, use a permutation of the functions on the reprogrammable I/Os identified in phases 2 and 3.

5.2.4.3 Note on Feature Disablement

GAP9 offers two registers, the purpose of which is to disable certain functionalities in an unalterable way: FEATURE_DISABLE and FEATURE_DISABLE_QK. These registers have the same behavior: the software may set to 1 a bit corresponding to a feature to disable, or to 0 for a feature which should be maintained; then the specific set-only bit 31 (DISABLE_LOCK) can be set to forbid any later update of the register value. However, those registers differ in the sense that FEATURE_DISABLE_QK is the SOC domain, and therefore is reset each time the SOC domain is switched off, whereas FEATURE_DISABLE retains its value during sleep modes, including deep sleep – only a hard reset or a power cycle resets its value.

The intended purpose of those registers is to be set early in the boot or wakeup process, so that their value is no more alterable when GAP9 has finished the software boot and starts the execution of the application. If desired, it is possible to force the configuration of the FEATURE_DISABLE register by the boot ROM, by setting the

FEATURE_DISABLE_SET bit of INFO_1 eFuse and setting FEATURE_DISABLE eFuse to target register values (see GAP9 ROM device description).

5.2.5 Advanced Timer/PWM Generator

The advanced timer component manages the following features:

- 4 advanced timers with 4 output signal channels each, providing PWM generation
- use of SOC clock or REF FAST clock as main timer clock, possibility to count on REF SLOW clock
- multiple trigger input sources:
  - output signal channels of all timers
  - 32 GPIOs
- configurable input trigger modes
- configurable prescaler for each timer
- configurable counting mode for each timer
- configurable channel threshold action for each timer
- 4 configurable output events
- configurable clock gating of each timer

5.2.5.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
T0_CMD	0x0	32	Timer 0 command
T0_CONFIG	0x4	32	Timer 0 configuration
T0_THRESHOLD	0x8	32	Timer 0 threshold configuration
T0_TH_CHANNEL0	0xC	32	Timer 0 channel 0 threshold configuration
T0_TH_CHANNEL1	0x10	32	Timer 0 channel 1 threshold configuration
T0_TH_CHANNEL2	0x14	32	Timer 0 channel 2 threshold configuration
T0_TH_CHANNEL3	0x18	32	Timer 0 channel 3 threshold configuration
T0_COUNTER	0x2C	32	Timer 0 counter
T1_CMD	0x40	32	Timer 1 command
T1_CONFIG	0x44	32	Timer 1 configuration
T1_THRESHOLD	0x48	32	Timer 1 threshold configuration
T1_TH_CHANNEL0	0x4C	32	Timer 1 channel 0 threshold configuration
T1_TH_CHANNEL1	0x50	32	Timer 1 channel 1 threshold configuration
T1_TH_CHANNEL2	0x54	32	Timer 1 channel 2 threshold configuration
T1_TH_CHANNEL3	0x58	32	Timer 1 channel 3 threshold configuration
T1_COUNTER	0x6C	32	Timer 1 counter
T2_CMD	0x80	32	Timer 2 command
T2_CONFIG	0x84	32	Timer 2 configuration
T2_THRESHOLD	0x88	32	Timer 2 threshold configuration
T2_TH_CHANNEL0	0x8C	32	Timer 2 channel 0 threshold configuration
T2_TH_CHANNEL1	0x90	32	Timer 2 channel 1 threshold configuration
T2_TH_CHANNEL2	0x94	32	Timer 2 channel 2 threshold configuration
T2_TH_CHANNEL3	0x98	32	Timer 2 channel 3 threshold configuration
T2_COUNTER	0xAC	32	Timer 2 counter
T3_CMD	0xC0	32	Timer 3 command
T3_CONFIG	0xC4	32	Timer 3 configuration
T3_THRESHOLD	0xC8	32	Timer 3 threshold configuration
T3_TH_CHANNEL0	0xCC	32	Timer 3 channel 0 threshold configuration

Name	Offset	Width	Description
T3_TH_CHANNEL1	0xD0	32	Timer 3 channel 1 threshold configuration
T3_TH_CHANNEL2	0xD4	32	Timer 3 channel 2 threshold configuration
T3_TH_CHANNEL3	0xD8	32	Timer 3 channel 3 threshold configuration
T3_COUNTER	0xEC	32	Timer 3 counter
EVENT_CFG	0x100	32	Output events configuration
CG	0x104	32	Channels clock gating configuration
CH_MUX	0x108	32	Channels output configuration

T0_CMD

Timer 0 command

Bit #	R/W	Name	Reset	Description
0	W	START	0x0	Write 1 to start timer 0
1	W	STOP	0x0	Write 1 to stop timer 0
2	W	UPDATE	0x0	Write 1 to update timer 0
3	W	RESET	0x0	Write 1 to reset timer 0
4	W	ARM	0x0	Write 1 to arm timer 0

T0_CONFIG

Timer 0 configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	INSEL	0x0	Timer 0 input source: 0-31: GPIO[0] to GPIO[31]; 32-35: Channel 0 to 3 of timer 0; 36-39: Channel 0 to 3 of timer 1; 40-43: Channel 0 to 3 of timer 2; 44-47: Channel 0 to 3 of timer 3
10:8	R/W	MODE	0x0	Timer 0 trigger mode: 0: trigger event at each clock cycle; 1: trigger event if input source is 0; 2: trigger event if input source is 1; 3: trigger event on input source rising edge; 4: trigger event on input source falling edge; 5: trigger event on input source both falling and rising edges; 6: trigger event on input source rising edge when armed; 7: trigger event on input source falling edge when armed
11	R/W	CLKSEL	0x0	Timer 0 counting clock: 0: main timer clock; 1: slow clock
12	R/W	UPDOWNSEL	0x1	timer 0 center-aligned mode: 0: the counter counts up and down alternatively; 1: the counter counts up and resets to 0 when it reaches the threshold
23:16	R/W	PRESC	0x0	Timer 0 prescaler value

T0_THRESHOLD

Timer 0 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH_LO	0x0	Timer 0 low threshold. It defines the start value of the counter
31:16	R/W	TH_HI	0x0	Timer 0 high threshold. It defines the end value of the counter

T0_TH_CHANNEL0

Timer 0 channel 0 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 0 channel 0 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 0 channel 0 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T0_TH_CHANNEL1

Timer 0 channel 1 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 0 channel 1 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 0 channel 1 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T0_TH_CHANNEL2

Timer 0 channel 2 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 0 channel 2 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 0 channel 2 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T0_TH_CHANNEL3

Timer 0 channel 3 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 0 channel 3 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 0 channel 3 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T0_COUNTER

Timer 0 counter

Bit #	R/W	Name	Reset	Description
15:0	R	COUNTER	0x0	Timer 0 counter value

T1_CMD

Timer 1 command

Bit #	R/W	Name	Reset	Description
0	R/W	START	0x0	Write 1 to start timer 1
1	R/W	STOP	0x0	Write 1 to stop timer 1
2	R/W	UPDATE	0x0	Write 1 to update timer 1
3	R/W	RESET	0x0	Write 1 to reset timer 1
4	R/W	ARM	0x0	Write 1 to arm timer 1

T1_CONFIG

Timer 1 configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	INSEL	0x0	Timer 1 input source: 0-31: GPIO[0] to GPIO[31]; 32-35: Channel 0 to 3 of timer 0; 36-39: Channel 0 to 3 of timer 1; 40-43: Channel 0 to 3 of timer 2; 44-47: Channel 0 to 3 of timer 3
10:8	R/W	MODE	0x0	Timer 1 trigger mode: 0: trigger event at each clock cycle; 1: trigger event if input source is 0; 2: trigger event if input source is 1; 3: trigger event on input source rising edge; 4: trigger event on input source falling edge; 5: trigger event on input source both falling and rising edges; 6: trigger event on input source rising edge when armed; 7: trigger event on input source falling edge when armed
11	R/W	CLKSEL	0x0	Timer 1 counting clock: 0: main timer clock; 1: slow clock
12	R/W	UPDOWNSEL	0x1	timer 1 center-aligned mode: 0: the counter counts up and down alternatively; 1: the counter counts up and resets to 0 when it reaches the threshold
23:16	R/W	PRESC	0x0	Timer 1 prescaler value

T1_THRESHOLD

Timer 1 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH_LO	0x0	Timer 1 low threshold. It defines the start value of the counter
31:16	R/W	TH_HI	0x0	Timer 1 high threshold. It defines the end value of the counter

T1_TH_CHANNEL0

Timer 1 channel 0 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 1 channel 0 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 1 channel 0 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T1_TH_CHANNEL1

Timer 1 channel 1 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 1 channel 1 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 1 channel 1 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T1_TH_CHANNEL2

Timer 1 channel 2 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 1 channel 2 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 1 channel 2 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T1_TH_CHANNEL3

Timer 1 channel 3 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 1 channel 3 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 1 channel 3 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T1_COUNTER

Timer 1 counter

Bit #	R/W	Name	Reset	Description
15:0	R	COUNTER	0x0	Timer 1 counter value

T2_CMD

Timer 2 command

Bit #	R/W	Name	Reset	Description
0	R/W	START	0x0	Write 1 to start timer 2
1	R/W	STOP	0x0	Write 1 to stop timer 2
2	R/W	UPDATE	0x0	Write 1 to update timer 2
3	R/W	RESET	0x0	Write 1 to reset timer 2
4	R/W	ARM	0x0	Write 1 to arm timer 2

T2_CONFIG

Timer 2 configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	INSEL	0x0	Timer 2 input source: 0-31: GPIO[0] to GPIO[31]; 32-35: Channel 0 to 3 of timer 0; 36-39: Channel 0 to 3 of timer 1; 40-43: Channel 0 to 3 of timer 2; 44-47: Channel 0 to 3 of timer 3
10:8	R/W	MODE	0x0	Timer 2 trigger mode: 0: trigger event at each clock cycle; 1: trigger event if input source is 0; 2: trigger event if input source is 1; 3: trigger event on input source rising edge; 4: trigger event on input source falling edge; 5: trigger event on input source both falling and rising edges; 6: trigger event on input source rising edge when armed; 7: trigger event on input source falling edge when armed
11	R/W	CLKSEL	0x0	Timer 2 counting clock: 0: main timer clock; 1: slow clock
12	R/W	UPDOWNSEL	0x1	Timer 2 center-aligned mode: 0: the counter counts up and down alternatively; 1: the counter counts up and resets to 0 when it reaches the threshold
23:16	R/W	PRESC	0x0	Timer 2 prescaler value

T2_THRESHOLD

Timer 2 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH_LO	0x0	Timer 2 low threshold. It defines the start value of the counter
31:16	R/W	TH_HI	0x0	Timer 2 high threshold. It defines the end value of the counter

T2_TH_CHANNEL0

Timer 2 channel 0 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 2 channel 0 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 2 channel 0 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T2_TH_CHANNEL1

Timer 2 channel 1 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 2 channel 1 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 2 channel 1 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T2_TH_CHANNEL2

Timer 2 channel 2 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 2 channel 2 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 2 channel 2 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T2_TH_CHANNEL3

Timer 2 channel 3 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 2 channel 3 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 2 channel 3 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T2_COUNTER

Timer 2 counter

Bit #	R/W	Name	Reset	Description
15:0	R	COUNTER	0x0	Timer 2 counter value

T3_CMD

Timer 3 command

Bit #	R/W	Name	Reset	Description
0	R/W	START	0x0	Write 1 to start timer 3
1	R/W	STOP	0x0	Write 1 to stop timer 3
2	R/W	UPDATE	0x0	Write 1 to update timer 3
3	R/W	RESET	0x0	Write 1 to reset timer 3
4	R/W	ARM	0x0	Write 1 to arm timer 3

T3_CONFIG

Timer 3 configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	INSEL	0x0	Timer 3 input source: 0-31: GPIO[0] to GPIO[31]; 32-35: Channel 0 to 3 of timer 0; 36-39: Channel 0 to 3 of timer 1; 40-43: Channel 0 to 3 of timer 2; 44-47: Channel 0 to 3 of timer 3
10:8	R/W	MODE	0x0	Timer 3 trigger mode: 0: trigger event at each clock cycle; 1: trigger event if input source is 0; 2: trigger event if input source is 1; 3: trigger event on input source rising edge; 4: trigger event on input source falling edge; 5: trigger event on input source both falling and rising edges; 6: trigger event on input source rising edge when armed; 7: trigger event on input source falling edge when armed
11	R/W	CLKSEL	0x0	Timer 3 counting clock: 0: main timer clock; 1: slow clock
12	R/W	UPDOWNSEL	0x1	Timer 3 center-aligned mode: 0: the counter counts up and down alternatively; 1: the counter counts up and resets to 0 when it reaches the threshold
23:16	R/W	PRESC	0x0	Timer 3 prescaler value

T3_THRESHOLD

Timer 3 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH_LO	0x0	Timer 3 low threshold. It defines the start value of the counter
31:16	R/W	TH_HI	0x0	Timer 3 high threshold. It defines the end value of the counter

T3_TH_CHANNEL0

Timer 3 channel 0 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 3 channel 0 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 3 channel 0 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T3_TH_CHANNEL1

Timer 3 channel 1 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 3 channel 1 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 3 channel 1 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T3_TH_CHANNEL2

Timer 3 channel 2 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 3 channel 2 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 3 channel 2 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T3_TH_CHANNEL3

Timer 3 channel 3 threshold configuration

Bit #	R/W	Name	Reset	Description
15:0	R/W	TH	0x0	Timer 3 channel 3 threshold
18:16	R/W	MODE	0x0	Action triggered on timer 3 channel 3 output when the channel threshold is reached: 0: set to 1; 1: toggle value at odd occurrence, clear to 0 at even occurrence; 2: set to 1 at odd occurrence, clear to 0 at even occurrence; 3: toggle value; 4: clear to 0; 5: toggle value at odd occurrence, set to 1 at even occurrence; 6: clear to 0 at odd occurrence, set to 1 at even occurrence

T3_COUNTER

Timer 3 counter

Bit #	R/W	Name	Reset	Description
15:0	R	COUNTER	0x0	Timer 3 counter value

EVENT_CFG

Output events configuration

Bit #	R/W	Name	Reset	Description
3:0	R/W	SEL0	0x0	Select source of output event 0: 0: timer 0 channel 0; 1: timer 0 channel 1; 2: timer 0 channel 2; 3: timer 0 channel 3; 4: timer 1 channel 0; 5: timer 1 channel 1; ... 14: timer 3 channel 2; 15: timer 3 channel 3
7:4	R/W	SEL1	0x0	Select source of output event 1: 0: timer 0 channel 0; 1: timer 0 channel 1; 2: timer 0 channel 2; 3: timer 0 channel 3; 4: timer 1 channel 0; 5: timer 1 channel 1; ... 14: timer 3 channel 2; 15: timer 3 channel 3
11:8	R/W	SEL2	0x0	Select source of output event 2: 0: timer 0 channel 0; 1: timer 0 channel 1; 2: timer 0 channel 2; 3: timer 0 channel 3; 4: timer 1 channel 0; 5: timer 1 channel 1; ... 14: timer 3 channel 2; 15: timer 3 channel 3
15:12	R/W	SEL3	0x0	Select source of output event 3: 0: timer 0 channel 0; 1: timer 0 channel 1; 2: timer 0 channel 2; 3: timer 0 channel 3; 4: timer 1 channel 0; 5: timer 1 channel 1; ... 14: timer 3 channel 2; 15: timer 3 channel 3
19:16	R/W	ENA	0x0	Output events enable: generation of event i is enabled if bit i=1

CG

Channels clock gating configuration

Bit #	R/W	Name	Reset	Description
3:0	R/W	ENA	0x0	Enable timers: bit i=0 gates the clock of timer i, bit i=1 enables timer i
7:4	R/W	MUX	0x0	Timers main clock selection: bit i=0: timer i uses SOC clock; bit i=1: timer i uses REF FAST clock

CH_MUX

Channels output configuration

Bit #	R/W	Name	Reset	Description
2:0	R/W	CH_SEL_0	0x0	Selects channel output sent to PWM0 output: 0 to 3: channel 0 to 3 of timer 0; 4 to 7: channel 0 to 3 of timer 1
5:3	R/W	CH_SEL_1	0x0	Selects channel output sent to PWM1 output: 0 to 3: channel 0 to 3 of timer 2; 4 to 7: channel 0 to 3 of timer 3
8:6	R/W	CH_SEL_2	0x0	Selects channel output sent to PWM2 output: 0 to 3: channel 0 to 3 of timer 0; 4 to 7: channel 0 to 3 of timer 1
11:9	R/W	CH_SEL_3	0x0	Selects channel output sent to PWM3 output: 0 to 3: channel 0 to 3 of timer 2; 4 to 7: channel 0 to 3 of timer 3
14:12	R/W	CH_SEL_4	0x0	Selects channel output sent to PWM4 output: 0 to 3: channel 0 to 3 of timer 0; 4 to 7: channel 0 to 3 of timer 1
17:15	R/W	CH_SEL_5	0x0	Selects channel output sent to PWM5 output: 0 to 3: channel 0 to 3 of timer 2; 4 to 7: channel 0 to 3 of timer 3
20:18	R/W	CH_SEL_6	0x0	Selects channel output sent to PWM6 output: 0 to 3: channel 0 to 3 of timer 0; 4 to 7: channel 0 to 3 of timer 1
23:21	R/W	CH_SEL_7	0x0	Selects channel output sent to PWM7 output: 0 to 3: channel 0 to 3 of timer 2; 4 to 7: channel 0 to 3 of timer 3

5.2.6 Soc Event Generator

SoC event generator component manages the following features:

- Soc events dispatching to FC event unit, Cluster event unit and uDMA
- FC High and Low timers input trigger events configuration
- SoC software event generation
- Event queue of width 2 for each event line with overflow error event generation
- 2 high priority events generation

Input events managed by SoC event generator are:

- 32kHz reference clock
- 48 SoC peripherals events inciming from uDMA interfaces, PMU, Advanced timers, GPIOs and RTC
- 8 software events

5.2.6.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
SW_EVENT	0x0	32	SoC software events trigger command register

Name	Offset	Width	Description
FC_MASK_0	0x4	32	FC event unit event dispatch mask configuration register for events 0 to 31
FC_MASK_1	0x8	32	FC event unit event dispatch mask configuration register for events 32 to 63
FC_MASK_2	0xC	32	FC event unit event dispatch mask configuration register for events 64 to 95
FC_MASK_3	0x10	32	FC event unit event dispatch mask configuration register for events 96 to 127
FC_MASK_4	0x14	32	FC event unit event dispatch mask configuration register for events 128 to 159
FC_MASK_5	0x18	32	FC event unit event dispatch mask configuration register for events 160 to 191
FC_MASK_6	0x1C	32	FC event unit event dispatch mask configuration register for events 192 to 223
FC_MASK_7	0x20	32	FC event unit event dispatch mask configuration register for events 224 to 255
CL_MASK_0	0x24	32	Cluster event dispatch mask configuration register for events 0 to 31
CL_MASK_1	0x28	32	Cluster event dispatch mask configuration register for events 32 to 63
CL_MASK_2	0x2C	32	Cluster event dispatch mask configuration register for events 64 to 95
CL_MASK_3	0x30	32	Cluster event dispatch mask configuration register for events 96 to 127
CL_MASK_4	0x34	32	Cluster event dispatch mask configuration register for events 128 to 159
CL_MASK_5	0x38	32	Cluster event dispatch mask configuration register for events 160 to 191
CL_MASK_6	0x3C	32	Cluster event dispatch mask configuration register for events 192 to 223
CL_MASK_7	0x40	32	Cluster event dispatch mask configuration register for events 224 to 255
PR_MASK_0	0x44	32	uDMA event dispatch mask configuration register for events 0 to 31
PR_MASK_1	0x48	32	uDMA event dispatch mask configuration register for events 32 to 63
PR_MASK_2	0x4C	32	uDMA event dispatch mask configuration register for events 64 to 95
PR_MASK_3	0x50	32	uDMA event dispatch mask configuration register for events 96 to 127
PR_MASK_4	0x54	32	uDMA event dispatch mask configuration register for events 128 to 159
PR_MASK_5	0x58	32	uDMA event dispatch mask configuration register for events 160 to 191
PR_MASK_6	0x5C	32	uDMA event dispatch mask configuration register for events 192 to 223
PR_MASK_7	0x60	32	uDMA event dispatch mask configuration register for events 224 to 255
ERR_0	0x64	32	Event queue overflow status register for events 0 to 31
ERR_1	0x68	32	Event queue overflow status register for events 32 to 63
ERR_2	0x6C	32	Event queue overflow status register for events 64 to 95
ERR_3	0x70	32	Event queue overflow status register for events 96 to 127
ERR_4	0x74	32	Event queue overflow status register for events 128 to 159
ERR_5	0x78	32	Event queue overflow status register for events 160 to 191

Name	Offset	Width	Description
ERR_6	0x7C	32	Event queue overflow status register for events 192 to 223
ERR_7	0x80	32	Event queue overflow status register for events 224 to 255
TIMER1_SEL_HI	0x84	32	FC High Timer1 source event configuration register
TIMER1_SEL_LO	0x88	32	FC Low Timer1 source event configuration register
TIMER2_SEL_HI	0x8C	32	FC High Timer2 source event configuration register
TIMER2_SEL_LO	0x90	32	FC Low Timer2 source event configuration register
FC_MASK_SET	0x94	32	Set the the FC mask of the specified event to 1
FC_MASK_CLR	0x98	32	Set the the FC mask of the specified event to 0

SW_EVENT

SoC software events trigger command register

Bit #	R/W	Name	Reset	Description
7:0	W	EVENT	0x0	Writing a one-hot value (bit *i*=1) into EVENT bitfield triggers SoC software event *i*. 8 software events are provided

FC_MASK_0

FC event unit event dispatch mask configuration register for events 0 to 31

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_1

FC event unit event dispatch mask configuration register for events 32 to 63

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_2

FC event unit event dispatch mask configuration register for events 64 to 95

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_3

FC event unit event dispatch mask configuration register for events 96 to 127

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_4

FC event unit event dispatch mask configuration register for events 128 to 159

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_5

FC event unit event dispatch mask configuration register for events 160 to 191

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_6

FC event unit event dispatch mask configuration register for events 192 to 223

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

FC_MASK_7

FC event unit event dispatch mask configuration register for events 224 to 255

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to FC event unit: setting bit i to 1 disables dispatching corresponding event to FC event unit; setting bit i to 0 enables dispatching corresponding event to FC event unit

CL_MASK_0

Cluster event dispatch mask configuration register for events 0 to 31

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_1

Cluster event dispatch mask configuration register for events 32 to 63

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_2

Cluster event dispatch mask configuration register for events 64 to 95

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_3

Cluster event dispatch mask configuration register for events 96 to 127

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_4

Cluster event dispatch mask configuration register for events 128 to 159

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_5

Cluster event dispatch mask configuration register for events 160 to 191

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_6

Cluster event dispatch mask configuration register for events 192 to 223

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

CL_MASK_7

Cluster event dispatch mask configuration register for events 224 to 255

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to Cluster event unit: setting bit i to 1 disables dispatching corresponding event to Cluster event unit; setting bit i to 0 enables dispatching corresponding event to Cluster event unit

PR_MASK_0

uDMA event dispatch mask configuration register for events 0 to 31

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_1

uDMA event dispatch mask configuration register for events 32 to 63

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_2

uDMA event dispatch mask configuration register for events 64 to 95

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_3

uDMA event dispatch mask configuration register for events 96 to 127

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_4

uDMA event dispatch mask configuration register for events 128 to 159

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_5

uDMA event dispatch mask configuration register for events 160 to 191

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_6

uDMA event dispatch mask configuration register for events 192 to 223

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

PR_MASK_7

uDMA event dispatch mask configuration register for events 224 to 255

Bit #	R/W	Name	Reset	Description
31:0	R/W	MASK	0xFFFFFFFF	Event mask to enable/disable event dispatch to uDMA peripherals: setting bit i to 1 disables dispatching corresponding event to uDMA; setting bit i to 0 enables dispatching corresponding event to uDMA

ERR_0

Event queue overflow status register for events 0 to 31

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_1

Event queue overflow status register for events 32 to 63

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_2

Event queue overflow status register for events 64 to 95

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_3

Event queue overflow status register for events 96 to 127

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_4

Event queue overflow status register for events 128 to 159

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_5

Event queue overflow status register for events 160 to 191

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_6

Event queue overflow status register for events 192 to 223

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

ERR_7

Event queue overflow status register for events 224 to 255

Bit #	R/W	Name	Reset	Description
31:0	R/W	ERR_VAL	0x0	Report event queues overflows: reading 1 on bit i means that an overflow occurred on the queue for corresponding SoC event. This field is cleared after read

TIMER1_SEL_HI

FC High Timer1 source event configuration register

Bit #	R/W	Name	Reset	Description
7:0	R/W	EVT	0x0	Configure which SoC event generator input event is propagated to FC Timer High trigger event input
31	R/W	ENA	0x0	Write 1 to enable SoC event generator event propagation to FC Timer High trigger event input

TIMER1_SEL_LO

FC Low Timer1 source event configuration register

Bit #	R/W	Name	Reset	Description
7:0	R/W	EVT	0x0	Configure which SoC event generator input event is propagated to FC Timer High trigger event input
31	R/W	ENA	0x0	Write 1 to enable SoC event generator event propagation to FC Timer High trigger event input

TIMER2_SEL_HI

FC High Timer2 source event configuration register

Bit #	R/W	Name	Reset	Description
7:0	R/W	EVT	0x0	Configure which SoC event generator input event is propagated to FC Timer High trigger event input
31	R/W	ENA	0x0	Write 1 to enable SoC event generator event propagation to FC Timer High trigger event input

TIMER2_SEL_LO

FC Low Timer2 source event configuration register

Bit #	R/W	Name	Reset	Description
7:0	R/W	EVT	0x0	Configure which SoC event generator input event is propagated to FC Timer High trigger event input
31	R/W	ENA	0x0	Write 1 to enable SoC event generator event propagation to FC Timer High trigger event input

FC_MASK_SET

Set the the FC mask of the specified event to 1

Bit #	R/W	Name	Reset	Description
7:0	W	EVENT	0x0	Write an event ID to set its FC mask to 1

FC_MASK_CLR

Set the the FC mask of the specified event to 0

Bit #	R/W	Name	Reset	Description
7:0	W	EVENT	0x0	Write an event ID to set its FC mask to 0

5.2.7 Power Management Unit

GAP9 Power Management Unit (PMU) is responsible for properly controlling GAP9 internal hardware to setup power modes, and for the transitions between those modes (see GAP9 Power Modes section for more information). This includes the control of the power network of GAP9 chip — start-up, wake-up and low power transition sequences — as well as the control of the reset of the power islands.

Configuration of PMU internal registers is done through a so-called PICL bus, using the DLCPD_* registers.

5.2.7.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
DLCPD_MSR	0x0	32	Maestro Status Register
DLCPD_MPACR	0x4	32	Maestro PICL Access Control Register
DLCPD_MPADR	0x8	32	Maestro PICL Access Data Register
DLCPD_IMR	0xC	32	Maestro Interrupt Mask Register
DLCPD_IFR	0x10	32	Maestro Interrupt Flag Register
DLCPD_IOK_IFR	0x14	32	Maestro ICU_OK Interrupt Flag Register
DLCPD_IDL_IFR	0x18	32	Maestro ICU_DELAYED Interrupt Flag Register
DLCPD_IDN_IFR	0x1C	32	Maestro ICU_DENIED Interrupt Flag Register
DLCPD_IUP_IFR	0x20	32	Maestro ICU_UPDATED Interrupt Flag Register

DLCPD_MSR

Maestro Status Register

Bit #	R/W	Name	Reset	Description
0	W	PICL_BUSY	0	PICL busy status. Set when a transfer is on going on the PICL bus. Cleared at the end of the PICL transfer.
1	W	SCU_BUSY	0	SCU busy status. Set when a SCU sequence is on going. Cleared when a SCU sequence ends.

DLCPD_MPACR

Maestro PICL Access Control Register

Bit #	R/W	Name	Reset	Description
15:0	R	PAADDR	0	PICL Access Address.
24	R	PADIR	0	PICL Access Direction: b0: write; b1: read.
28	R	PASTART	0	PICL Access Start. This field is automatically cleared when PICL access is finished.

DLCPD_MPADR

Maestro PICL Access Data Register

Bit #	R/W	Name	Reset	Description
15:0	W	PRWDATA	0	PICL Read/Write Data. This field is automatically updated after a PICL read access is finished.

DLCPD_IMR

Maestro Interrupt Mask Register

Bit #	R/W	Name	Reset	Description
0	W	ICU_OK_M	0	Mask of ICU_OK interrupt.
1	W	ICU_DLY_M	0	Mask of ICU_DELAYED interrupt.
2	W	ICU_DEN_M	0	Mask of ICU_DENIED interrupt.
3	W	ICU_UPD_M	0	Mask of ICU_UPDATED interrupt.
6	W	PICL_OK_M	0	Mask of PICL_OK interrupt.
7	W	SCU_OK_M	0	Mask of SCU_OK interrupt.
8	W	SCU_FL_M	0	Mask of SCU_FL interrupt.

DLCPD_IFR

Maestro Interrupt Flag Register

Bit #	R/W	Name	Reset	Description
0	W	ICU_OK_F	0	Set when at least one of the bit of the DLCPD_IOK_IFR register is set.
1	W	ICU_DLY_F	0	Set when at least one of the bit of the DLCPD_IDL_IFR register is set.
2	W	ICU_DEN_F	0	Set when at least one of the bit of the DLCPD_IDN_IFR register is set.
3	W	ICU_UPD_F	0	Set when at least one of the bit of the DLCPD_IUP_IFR register is set.
6	W	PICL_OK_F	0	Set when PICL transfer is finished. Cleared when writing 1 in this field.
7	W	SCU_OK_F	0	Set when SCU sequence is finished without error. Cleared when writing 1 in this field.
8	W	SCU_FL_F	0	Set when SCU sequence is finished with error. Cleared when writing 1 in this field.

DLCPD_IOK_IFR

Maestro ICU_OK Interrupt Flag Register

Bit #	R/W	Name	Reset	Description
31:0	W	ICU_OK_FLAGS	0	Flags of the ICU_OK interrupts. Each bit is set if the requested mode change from the control interface on the corresponding ICU was performed. Each bit is cleared when writing it to 1.

DLCPD_IDL_IFR

Maestro ICU_DELAYED Interrupt Flag Register

Bit #	R/W	Name	Reset	Description
31:0	W	ICU_DLY_FLAGS	0	Flags of the ICU_DELAYED interrupts. Each bit is set if the requested mode change from the control interface on the corresponding ICU was delayed. Each bit is cleared when writing it to 1.

DLCPD_IDN_IFR

Maestro ICU_DENIED Interrupt Flag Register

Bit #	R/W	Name	Reset	Description
31:0	W	ICU_DEN_FLAGS	0	Flags of the ICU_DENIED interrupts. Each bit is set if the requested mode change from the control interface on the corresponding ICU was denied. Each bit is cleared when writing it to 1.

DLCPD_IUP_IFR

Maestro ICU_UPDATED Interrupt Flag Register

Bit #	R/W	Name	Reset	Description
31:0	W	ICU_UPD_FLAGS	0	Flags of the ICU_UPDATED interrupts. Each bit is set when the corresponding ICU changed its mode or order. Each bit is cleared when writing it to 1.

5.2.8 Real-Time Clock

RTC is an always-operating function used to keep track of "time of the day", even when the most of the device is off. It can also be used to wake up the device from low power modes on regular intervals. RTC internal registers are indirectly accessed through an APB bridge, controlled by the APB_* registers below.

Note: The RTC is available only when the REF SLOW clock is available.

5.2.8.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
APB_SR	0x0	32	RTC APB status register
APB_CR	0x4	32	RTC APB control register
APB_DR	0x8	32	RTC APB data register
APB_ICR	0x10	32	RTC APB interruption control register
APB_IMR	0x14	32	RTC APB interruption mask register
APB_IFR	0x18	32	RTC APB interruption flag register

APB_SR

RTC APB status register

Bit #	R/W	Name	Reset	Description
1:0	R	APB_INT	0x0	APB interrupt status bitfield: b0: no interruption has been requested; b1: interruption requested

APB_CR

RTC APB control register

Bit #	R/W	Name	Reset	Description
5:0	R/W	APB_ADDR	0x0	Indirect access address
16	R/W	APB_OP	0x0	Indirect access operation: b0: APB read operation; b1: APB write operation

APB_DR

RTC APB data register

Bit #	R/W	Name	Reset	Description
31:0	R/W	APB_DATA	0x0	Indirect access data

APB_ICR

RTC APB interruption control register

Bit #	R/W	Name	Reset	Description
1:0	R/W	MODE	0x0	APB interrupt signal mode configuration: b00: APB interrupt is a high level signal; b01: APB interrupt is a low level signal; b10: APB interrupt is a high level pulse of 1 REF SLOW clock period; b11: APB interrupt is a low level pulse of 1 REF SLOW clock period

APB_IMR

RTC APB interruption mask register

Bit #	R/W	Name	Reset	Description
0	R/W	READ_MASK	0x0	APB read operation interruption mask: b0: enabled; b1: disabled
1	R/W	WRITE_MASK	0x0	APB write operation interruption mask: b0: enabled; b1: disabled

APB_IFR

RTC APB interruption flag register

Bit #	R/W	Name	Reset	Description
0	R/W	READ_FLAG	0x0	APB read operation status flag: b0: nothing; b1: read operation done and requested indirect data is available
1	R/W	WRITE_FLAG	0x0	APB write operation status flag: b0: nothing; b1: write operation done

5.2.8.2 Register map for indirectly accessed registers

Overview

Name	Offset	Width	Description
RTC_SR	0x0	32	RTC status register
RTC_CR	0x1	32	RTC control register
RTC_ICR	0x8	32	RTC interrupt control register
RTC_IMR	0x9	32	RTC interrupt mask register
RTC_IFR	0xA	32	RTC interrupt flag register
CALENDAR_CONTROL	0x10	32	RTC calendar control register
CALENDAR_TIME	0x12	32	RTC calendar time register
CALENDAR_DATE	0x13	32	RTC calendar date register
ALARM_CONTROL	0x18	32	RTC alarm control register
ALARM1_TIME	0x1A	32	RTC alarm 1 time register
ALARM1_DATE	0x1B	32	RTC alarm 1 date register
COUNTDOWN_CONTROL	0x20	32	RTC countdown control register
COUNTDOWN1_INIT	0x21	32	RTC countdown 1 initialisation register
COUNTDOWN1_TIMER	0x22	32	RTC countdown 1 timer register
CKIN_DIV1	0x28	32	RTC clock divider 1 register
CKREF_CONF	0x2A	32	RTC reference clock configuration register
RTC_TEST_REG_A	0x30	32	RTC test register

RTC_SR

RTC status register

Bit #	R/W	Name	Reset	Description
0	R	INT	0x0	RTC interrupt status: b0: no interruption has been requested; b1: interruption requested

RTC_CR

RTC control register

Bit #	R/W	Name	Reset	Description
0	R/W	SB	0x0	RTC standby configuration: b0: RTC is in active mode; b1: RTC is in standby mode
4	R/W	CAL_EN	0x0	Calibration process activation (unused, must remain b0)
8	R/W	SOFT_RST	0x0	Soft reset command: b0: no reset; b1: reset the calendar, alarm, countdown and clock generation features and associated registers (CALENDAR*, ALARM*, COUNTDOWN*, CKIN_DIV* and CKREF_CONF). In single mode, this bitfield is automatically updated to b1 after the event occurs.

RTC_ICR

RTC interrupt control register

Bit #	R/W	Name	Reset	Description
1:0	R/W	MODE	0x0	RTC interrupt signal mode configuration. Sets the state of the signal when an RTC interrupt occurs: b00: signal set high; b01: signal set low; b10: pulse high during 1 REF SLOW clock period; b11: pulse low during 1 REF SLOW clock period

RTC_IMR

RTC interrupt mask register

Bit #	R/W	Name	Reset	Description
0	R/W	ALARM_MASK	0x1	Alarm 1 interrupt mask configuration: b0: interrupt enabled; b1: interrupt disabled
4	R/W	TIMER_MASK	0x1	Timer 1 interrupt mask configuration: b0: interrupt enabled; b1: interrupt disabled
12	R/W	CAL_MASK	0x1	Calibration interrupt mask configuration (unused): b0: interrupt enabled; b1: interrupt disabled

RTC_IFR

RTC interrupt flag register

Bit #	R/W	Name	Reset	Description
0	R	ALARM_FLAG	0x0	Alarm 1 interrupt status flag: b0: no interrupt; b1: calendar has reached the date and time set in alarm 1
4	R	TIMER_FLAG	0x0	Timer 1 interrupt status flag: b0: no interrupt; b1: countdown timer 1 has reached value 0
12	R	CAL_FLAG	0x0	Calibration interrupt status flag (unused): b0: nothing; b1: calibration process has ended

CALENDAR_CONTROL

RTC calendar control register

Bit #	R/W	Name	Reset	Description
0	R/W	SB	0x0	Calendar standby configuration: b0: calendar is active; b1: calendar is disabled

CALENDAR_TIME

RTC calendar time register

Bit #	R/W	Name	Reset	Description
3:0	R/W	SEC0	0x0	Calendar time: seconds digit 0
7:4	R/W	SEC1	0x0	Calendar time: seconds digit 1
11:8	R/W	MIN0	0x0	Calendar time: minutes digit 0
15:12	R/W	MIN1	0x0	Calendar time: minutes digit 1
19:16	R/W	HOU0	0x0	Calendar time: hours digit 0
23:20	R/W	HOU1	0x0	Calendar time: hours digit 1

CALENDAR_DATE

RTC calendar date register

Bit #	R/W	Name	Reset	Description
3:0	R/W	DAY0	0x1	Calendar date: day digit 0
7:4	R/W	DAY1	0x0	Calendar date: day digit 1
11:8	R/W	MON0	0x1	Calendar date: month digit 0
15:12	R/W	MON1	0x0	Calendar date: month digit 1
19:16	R/W	YEA0	0x0	Calendar date: year digit 0
23:20	R/W	YEA1	0x0	Calendar date: year digit 1

ALARM_CONTROL

RTC alarm control register

Bit #	R/W	Name	Reset	Description
0	R/W	SB	0x1	Alarm 1 standby configuration: b0: alarm is active; b1: alarm is inactive. In single mode, this bitfield is automatically updated to b1 after the event occurs
4	R/W	MODE	0x0	Alarm 1 mode configuration: b0: single; b1: repeat
19:16	R/W	CFG	0x6	Alarm 1 repeat configuration: b0011: every second; b0100: every minute; b0101: every hour; b0110: every day; b0111: every month; b1000: every year

ALARM1_TIME

RTC alarm 1 time register

Bit #	R/W	Name	Reset	Description
3:0	R/W	SEC0	0x0	Alarm 1 time: seconds digit 0
7:4	R/W	SEC1	0x0	Alarm 1 time: seconds digit 1
11:8	R/W	MIN0	0x0	Alarm 1 time: minutes digit 0
15:12	R/W	MIN1	0x0	Alarm 1 time: minutes digit 1
19:16	R/W	HOU0	0x0	Alarm 1 time: hours digit 0
23:20	R/W	HOU1	0x0	Alarm 1 time: hours digit 1

ALARM1_DATE

RTC alarm 1 date register

Bit #	R/W	Name	Reset	Description
3:0	R/W	DAY0	0x0	Alarm 1 date: day digit 0
7:4	R/W	DAY1	0x0	Alarm 1 date: day digit 1
11:8	R/W	MON0	0x0	Alarm 1 date: month digit 0
15:12	R/W	MON1	0x0	Alarm 1 date: month digit 1
19:16	R/W	YEA0	0x0	Alarm 1 date: year digit 0
23:20	R/W	YEA1	0x0	Alarm 1 date: year digit 1

COUNTDOWN_CONTROL

RTC countdown control register

Bit #	R/W	Name	Reset	Description
0	R/W	SB	0x1	Countdown timer 1 standby configuration: b0: countdown timer is active; b1: countdown timer is inactive. In single mode, this bitfield is automatically updated to b1 after the event occurs
4	R/W	MODE	0x0	Countdown timer 1 mode configuration: b0: single; b1: repeat

COUNTDOWN1_INIT

RTC countdown 1 initialisation register

Bit #	R/W	Name	Reset	Description
31:0	R/W	VAL	0xFFFFFFFF	Countdown timer 1 initial value

COUNTDOWN1_TIMER

RTC countdown 1 timer register

Bit #	R/W	Name	Reset	Description
31:0	R	VAL	0x00000000	Countdown timer 1 current value

CKIN_DIV1

RTC clock divider 1 register

Bit #	R/W	Name	Reset	Description
15:0	R/W	VAL	0x8000	Countdown timer 1 clock divider value

CKREF_CONF

RTC reference clock configuration register

Bit #	R/W	Name	Reset	Description
21:0	R/W	VAL	0x0F4240	Countdown 1 reference clock configuration

RTC_TEST_REG_A

RTC test register

Bit #	R/W	Name	Reset	Description
21:0	R/W	RESERVED	0x00001E3D	Reserved for test

5.2.9 FC I-Cache Controller

The FC instruction cache controller offers the following features:

- Bypassable instruction cache
- Flush and selective flush commands
- Instruction cache pending action status flag

5.2.9.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
ENABLE	0x0	32	FC instruction cache unit enable configuration register.
FLUSH	0x4	32	FC instruction cache unit flush command register.
SEL_FLUSH	0x8	32	FC instruction cache unit selective flush command register.
STATUS	0xC	32	FC instruction cache unit status register.

ENABLE

FC instruction cache unit enable configuration register.

Bit #	R/W	Name	Reset	Description
0	W	ENABLE	0x0	Enable FC instruction cache: b0: disabled; b1: enabled

FLUSH

FC instruction cache unit flush command register.

Bit #	R/W	Name	Reset	Description
0	W	FLUSH	0x0	Write 1 to fully flush the FC instruction cache

SEL_FLUSH

FC instruction cache unit selective flush command register.

Bit #	R/W	Name	Reset	Description
31:0	W	SEL_FLUSH	0x0	Write an address to selectively flush it

STATUS

FC instruction cache unit status register.

Bit #	R/W	Name	Reset	Description
0	R	STATUS	0x0	Pending action status flag: b0: no pending caching action; b1: pending caching action

5.2.10 FC Interrupt Controller

FC interrupt controller manages the following features:

- Enable/disable interrupt
- Check interrupt status

5.2.10.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
MASK	0x0	32	Mask status register.
MASK_SET	0x4	32	Mask set register.
MASK_CLEAR	0x8	32	Mask clear register.
STATUS	0xC	32	Interrupt status register.
STATUS_SET	0x10	32	Interrupt enable register.
STATUS_CLEAR	0x14	32	Interrupt clear register.
FIFO	0x24	32	Read access to SoC events FIFO.

MASK

Mask status register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	IT_MASK	0x0	Interrupt mask: set bit i to 1 to enable interrupt i

MASK_SET

Mask set register.

Bit #	R/W	Name	Reset	Description
31:0	W	IT_MASK_SET	0x0	Write 1 to bit i to set bit i of MASK register

MASK_CLEAR

Mask clear register.

Bit #	R/W	Name	Reset	Description
31:0	W	IT_MASK_CLEAR	0x0	Write 1 to bit i to clear bit i of MASK register

STATUS

Interrupt status register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	IT_STATUS	0x0	Interrupt status: bit i is 1 when interrupt i is active

STATUS_SET

Interrupt enable register.

Bit #	R/W	Name	Reset	Description
31:0	W	IT_STATUS_SET	0x0	Write 1 to bit i to set bit i of STATUS register

STATUS_CLEAR

Interrupt clear register.

Bit #	R/W	Name	Reset	Description
31:0	W	IT_STATUS_CLEAR	0x0	Write 1 to bit i to clear bit i of STATUS register

FIFO

Read access to SoC events FIFO.

Bit #	R/W	Name	Reset	Description
31:0	R	EVENT_NUM	0x0	Reading this field pops an event from SoC event FIFO and returns the event ID

5.2.11 I3C Interface

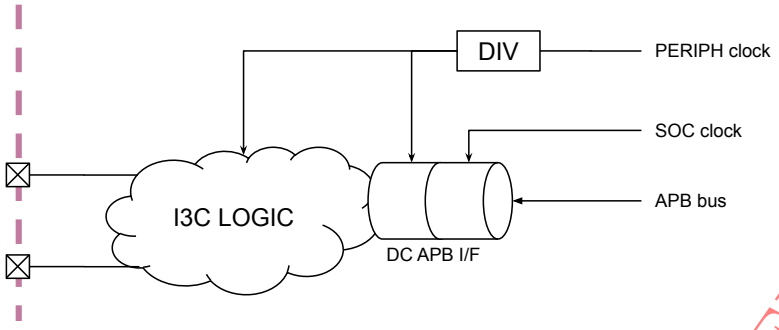
GAP9 I3C interface manages the following features:

- Controls all I3C bus specific sequencing, protocol, arbitration and timing.
- Configurable I3C clock frequency related to SoC clock frequency.
- Status flags for busy bus and arbitration lost.

I3C interface uses a stream pre-processing protocol to ease the construction of I3C transfers combining commands and data stream.

5.2.11.1 Clocking

The figure below shows the clocking scheme of the I3C peripheral. The peripheral is clocked by the PERIPH clock generated by the FLL, on which an integer division may be applied. Note that the I3C configuration interface – not shown on the figure – is clocked by SOC clock.



5.2.11.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
Mst_Cntl_En_Reg	0x0	32	I3C Master control.
CMD_Tr_Req_Reg_1	0x4	32	MSBs of command FIFO: data length, and data byte value to be sent along with the CCC or HDR command.
CMD_Tr_Req_Reg_2	0x8	32	LSBs of command FIFO: command type, Transaction ID, CCC code and slave address. Writing to this register triggers command execution.
Resp_Fifo_Reg	0x10	32	FIFO for received responses: success or failure of the command, transaction ID and the remaining data. Up to 32 outstanding responses can be stored.
IBI_Resp_Reg	0x14	32	In-Band interrupt status, timestamping information present, dynamic address of the slave or Hot Join request.
IBI_Data_Reg	0x18	32	Data received from the slave during the IBI process.
Data_RX_FIFO_Reg	0x1C	32	Received data FIFO (Data_RX_FIFO).
Data_TX_FIFO_Reg	0x20	32	Sending data FIFO (Data_TX_FIFO).
IRQ_STATUS_Reg	0x30	32	Status of the event happened during the transfer.
TCAS_TIMER_Reg	0x40	32	Timing for Clock after Start condition (TCAS) is the time (in Number of Reference clocks) after the start condition after which the SCL pin can go low.
TLOW_OD_TIMER_Reg	0x44	32	Low period of SCL pin in Open-Drain mode (in Number of Reference clock) after the SCL pin can go high.
THIGH_OD_TIMER_Reg	0x48	32	High period of SCL pin in Open-Drain mode (in Number of Reference clock) after the SCL pin can go low.
TLOW_PP_TIMER_Reg	0x4C	32	Low period of SCL pin in Push-Pull mode (in Number of Reference clock) after the SCL pin can go high.
THIGH_PP_TIMER_Reg	0x50	32	High period of SCL pin in Push-Pull mode (in Number of Reference clock) after the SCL pin can go low.
TDS_TIMER_Reg	0x54	32	SDA data setup time during both Open-Drain/Push-Pull mode (in Number of Reference clock).
THD_PP_TIMER_Reg	0x58	32	SDA data hold time during the Push-Pull mode (in Number of Reference clock).
TCBP_TIMER_Reg	0x5C	32	Clock time before Stop condition.
TCBSR_TIMER_Reg	0x60	32	Clock time before Repeated start condition.
THD_DDR_TIMER_Reg	0x64	32	SDA data hold time during the Push-Pull mode (in Number of Reference clock) in DDR Data rate.
BUS_FREE_TIMER_Reg	0x68	32	Bus free time between the Stop condition and the next start condition (in Number of Reference clock).

Name	Offset	Width	Description
BUS_AVAIL_TIMER_Reg	0x6C	32	Time to keep the SDA and SCL pin to High (in Number of Reference clock).
TIDLE_TIMER_Reg	0x70	32	Extended duration of the bus free condition after the Stop condition (in Number of Reference clock) to enable the device to drive the Hot Join request.
TSCO_TIMER_Reg	0x74	32	Maximum time the slave needs to drive the bus during the ACK/read data after the clock change.
TSU_STA_TIMER_Reg	0x90	32	SDA data setup time during both Open-Drain (in Number of Reference clock) for a Repeated Start.
THD_STA_TIMER_Reg	0x94	32	SDA data hold time during the Open Drain mode (in Number of Reference clock).
TLOW_TIMER_Reg	0x98	32	Low period of SCL pin in Open Drain mode during Legacy I2C Mode (in Number of Reference clock) after the timer reached the SCL pin can go high.
THIGH_TIMER_Reg	0x9C	32	High period of SCL pin in Open Drain Mode for Legacy I2C (in Number of Reference clock) after this timer count reached the SCL pin can go low.
TVD_DATA_TIMER_Reg	0xA0	32	Data hold time in Open Drain Mode for Legacy I2C (in Number of Reference clock) after this timer count reached the SDA pin can change its value.
TSU_STOP_TIMER_Reg	0xA8	32	SDA data setup time during Open-Drain (in Number of Reference clock) for Stop condition.
Device_Addr_Table_Reg0	0x100	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave0.
Device_Addr_Table_Reg1	0x104	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave1.
Device_Addr_Table_Reg2	0x108	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave2.
Device_Addr_Table_Reg3	0x10C	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave3.
Device_Addr_Table_Reg4	0x110	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave4.
Device_Addr_Table_Reg5	0x114	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave5.
Device_Addr_Table_Reg6	0x118	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave6.
Device_Addr_Table_Reg7	0x11C	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave7.
Device_Addr_Table_Reg8	0x120	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave8.
Device_Addr_Table_Reg9	0x124	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave9.
Device_Addr_Table_Reg10	0x128	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave10.
Device_Addr_Table_Reg11	0x12C	32	Device type, IBI handling and Dynamic address to be set and Static address of the slave11.
Device_Char_Table_Reg0_0	0x200	32	MSB of PID value of slave0 during the dynamic address assignment command.
Device_Char_Table_Reg1_0	0x204	32	LSB of PID value of slave0 during the dynamic address assignment command.
Device_Char_Table_Reg2_0	0x208	32	BCR, DCR and dynamic address of slave0 during the dynamic address assignment command.
Device_Char_Table_Reg0_1	0x210	32	MSB of PID value of slave1 during the dynamic address assignment command.

Name	Offset	Width	Description
Device_Char_Table_-Reg1_1	0x214	32	LSB of PID value of slave1 during the dynamic address assignment command.
Device_Char_Table_-Reg2_1	0x218	32	BCR, DCR and dynamic address of slave1 during the dynamic address assignment command.
Device_Char_Table_-Reg0_2	0x220	32	MSB of PID value of slave2 during the dynamic address assignment command.
Device_Char_Table_-Reg1_2	0x224	32	LSB of PID value of slave2 during the dynamic address assignment command.
Device_Char_Table_-Reg2_2	0x228	32	BCR, DCR and dynamic address of slave2 during the dynamic address assignment command.
Device_Char_Table_-Reg0_3	0x230	32	MSB of PID value of slave3 during the dynamic address assignment command.
Device_Char_Table_-Reg1_3	0x234	32	LSB of PID value of slave3 during the dynamic address assignment command.
Device_Char_Table_-Reg2_3	0x238	32	BCR, DCR and dynamic address of slave3 during the dynamic address assignment command.
Device_Char_Table_-Reg0_4	0x240	32	MSB of PID value of slave4 during the dynamic address assignment command.
Device_Char_Table_-Reg1_4	0x244	32	LSB of PID value of slave4 during the dynamic address assignment command.
Device_Char_Table_-Reg2_4	0x248	32	BCR, DCR and dynamic address of slave4 during the dynamic address assignment command.
Device_Char_Table_-Reg0_5	0x250	32	MSB of PID value of slave5 during the dynamic address assignment command.
Device_Char_Table_-Reg1_5	0x254	32	LSB of PID value of slave5 during the dynamic address assignment command.
Device_Char_Table_-Reg2_5	0x258	32	BCR, DCR and dynamic address of slave5 during the dynamic address assignment command.
Device_Char_Table_-Reg0_6	0x260	32	MSB of PID value of slave6 during the dynamic address assignment command.
Device_Char_Table_-Reg1_6	0x264	32	LSB of PID value of slave6 during the dynamic address assignment command.
Device_Char_Table_-Reg2_6	0x268	32	BCR, DCR and dynamic address of slave6 during the dynamic address assignment command.
Device_Char_Table_-Reg0_7	0x270	32	MSB of PID value of slave7 during the dynamic address assignment command.
Device_Char_Table_-Reg1_7	0x274	32	LSB of PID value of slave7 during the dynamic address assignment command.
Device_Char_Table_-Reg2_7	0x278	32	BCR, DCR and dynamic address of slave7 during the dynamic address assignment command.
Device_Char_Table_-Reg0_8	0x280	32	MSB of PID value of slave8 during the dynamic address assignment command.
Device_Char_Table_-Reg1_8	0x284	32	LSB of PID value of slave8 during the dynamic address assignment command.
Device_Char_Table_-Reg2_8	0x288	32	BCR, DCR and dynamic address of slave8 during the dynamic address assignment command.
Device_Char_Table_-Reg0_9	0x290	32	MSB of PID value of slave9 during the dynamic address assignment command.
Device_Char_Table_-Reg1_9	0x294	32	LSB of PID value of slave9 during the dynamic address assignment command.
Device_Char_Table_-Reg2_9	0x298	32	BCR, DCR and dynamic address of slave9 during the dynamic address assignment command.
Device_Char_Table_-Reg0_10	0x2A0	32	MSB of PID value of slave10 during the dynamic address assignment command.

Name	Offset	Width	Description
Device_Char_Table_-Reg1_10	0x2A4	32	LSB of PID value of slave10 during the dynamic address assignment command.
Device_Char_Table_-Reg2_10	0x2A8	32	BCR, DCR and dynamic address of slave10 during the dynamic address assignment command.
Device_Char_Table_-Reg0_11	0x2B0	32	MSB of PID value of slave11 during the dynamic address assignment command.
Device_Char_Table_-Reg1_11	0x2B4	32	LSB of PID value of slave11 during the dynamic address assignment command.
Device_Char_Table_-Reg2_11	0x2B8	32	BCR, DCR and dynamic address of slave11 during the dynamic address assignment command.

Mst_Cntl_En_Reg

I3C Master control.

Bit #	R/W	Name	Reset	Description
0	R/W	MASTER_CONTROL_EN	0x0	Enable I3C master interface: b0: disabled; b1: enabled
1	R/W	MASTER_RST_EN	0x0	Enable master controller to issue soft reset during any transfer. Soft reset will be applied after the completion of ACK/NACK state. b0: disabled; b1: enabled

CMD_Tr_Req_Reg_1

MSBs of command FIFO: data length, and data byte value to be sent along with the CCC or HDR command.

Bit #	R/W	Name	Reset	Description
31:0	W	DATA	0x0	Data length or data byte value, depending on the type of command being initiated. See description of commands.

CMD_Tr_Req_Reg_2

LSBs of command FIFO: command type, Transaction ID, CCC code and slave address. Writing to this register triggers command execution.

Bit #	R/W	Name	Reset	Description
31:0	W	CMD_REQ	0x0	Command request, which contains information about the command type, transaction ID, slave address and the type of CCC or HDR to be initiated. See description of commands.

Resp_Fifo_Reg

FIFO for received responses: success or failure of the command, transaction ID and the remaining data. Up to 32 outstanding responses can be stored.

Bit #	R/W	Name	Reset	Description
15:0	R	DATA_LEN	0x0	Write transfer: remaining Data length (in bytes); Read transfer: number of read data received from the slave; Address command: remaining device count
27:24	R	TID	0x0	Command request Tag ID
31:28	R	ERR_STATUS	0x0	Type of error or success for the command transfer: 0x0: Success; 0x1: CRC Error; 0x2: Parity Error; 0x3-0x4: Reserved; 0x5: NACK error for Dynamic address, M0/M2 Error & HDR NACK error; 0x6: Overflow/underflow error; 0x7: Reserved; 0x8: Data success with Retry operation; 0x9-0xD: Reserved; 0xE: Overflow/underflow error with Retry operation; 0xF: Reserved

IBI_Resp_Reg

In-Band interrupt status, timestamping information present, dynamic address of the slave or Hot Join request.

Bit #	R/W	Name	Reset	Description
7:0	R	DATA_LEN	0x0	IBI Data length. Number of bytes in the IBI requests received.
15:8	R	IBI_ID	0x0	IBI received ID. Contains slave address for IBI. Contains Hot Join ID for the Hot-Join IBI
25	R	TS_PRESENT	0x0	IBI timestamp present for IBI: b1: IBI is timestamped; b0: IBI is not timestamped
30:26	R/W	RESERVED_5	0x0	Reserved/Not used.
31	R	IBI_STS	0x0	IBI status. Indicates how the IBI is handled: b0: Indicates IBI is handled with ACK; b1: Indicates IBI is handled with NACK

IBI_Data_Reg

Data received from the slave during the IBI process.

Bit #	R/W	Name	Reset	Description
31:0	R	IBI_DATA	0x0	IBI data. Data received during the IBI process is stored in FIFO and is sent out to the software using this register.

Data_RX_FIFO_Reg

Received data FIFO (Data_RX_FIFO).

Bit #	R/W	Name	Reset	Description
31:0	R	DATA_RX	0x0	All the data received from the slave is stored in a FIFO. For each read of this register, a value are retrieved from the FIFO.

Data_TX_FIFO_Reg

Sending data FIFO (Data_TX_FIFO).

Bit #	R/W	Name	Reset	Description
31:0	W	DATA_TX	0x0	Write data to be sent from the master to the slave. Software writes this register to send the write data to the slave.

IRQ_STATUS_Reg

Status of the event happened during the transfer.

Bit #	R/W	Name	Reset	Description
0	R	RESP_DONE	0x0	Indicate the response completion is done.
1	R	DATA_TX_FIFO_-FULL	0x0	Indicate the data TX FIFO is full.
2	R	DATA_TX_FIFO_-EMPTY	0x0	Indicate the data TX FIFO is empty.
3	R	CMD_REQ_FIFO_-FULL	0x0	Indicate the command transfer request FIFO Full.
4	R	DATA_RX_FIFO_-FULL	0x0	Indicate the data RX FIFO Full.
5	R	RESP_COMPL	0x0	Indicate the response completion FIFO is full.
6	R	IBI_TRANSFER_-DONE	0x0	Indicate the IBI request is received from slave.
7	R	IBI_DATA_RX_-FIFO_FULL	0x0	Indicate the IBI payload is full.
8	R	RST_COMPLETION	0x0	Indicate the Master controller issued soft reset.

TCAS_TIMER_Reg

Timing for Clock after Start condition (TCAS) is the time (in Number of Reference clocks) after the start condition after which the SCL pin can go low.

Bit #	R/W	Name	Reset	Description
31:0	R/W	TCAS_TIMER	0x3	Timing of the SCL pin to go low after the start condition (number of clock cycles). As per specification, min time value is 38.4ns. For example, with a 100MHz clock, 4 clock periods are required. So, the value of the register is 0x3.

TLOW_OD_TIMER_Reg

Low period of SCL pin in Open-Drain mode (in Number of Reference clock) after the SCL pin can go high.

Bit #	R/W	Name	Reset	Description
4:0	R/W	TLOW_OD_TIMER	0x23	Low period of the SCL clock pin (number of clock cycles). As per specification, min time value is 200ns + fall time of SDA signal. For example, with a 100MHz clock, 24 clock periods are required. So, the value of the register is 0x23.

THIGH_OD_TIMER_Reg

High period of SCL pin in Open-Drain mode (in Number of Reference clock) after the SCL pin can go low.

Bit #	R/W	Name	Reset	Description
4:0	R/W	THIGH_OD_TIMER	0x2	High period of the SCL clock pin (number of clock cycles). As per specification, max time value is 41ns. For example, with a 100MHz clock, max. of 2 clock periods is allowed. So, the value of the register is 0x2.

TLOW_PP_TIMER_Reg

Low period of SCL pin in Push-Pull mode (in Number of Reference clock) after the SCL pin can go high.

Bit #	R/W	Name	Reset	Description
4:0	R/W	TLOW_PP_TIMER	0x3	Low period of the SCL clock pin (number of clock cycles) in push-pull mode. As per specification, min time value is 24ns. For example, with a 100MHz clock, 3 clock periods are required. So, the value of the register is 0x3.

THIGH_PP_TIMER_Reg

High period of SCL pin in Push-Pull mode (in Number of Reference clock) after the SCL pin can go low.

Bit #	R/W	Name	Reset	Description
4:0	R/W	THIGH_PP_TIMER	0x2	High period of the SCL clock pin (number of clock cycles). As per specification, max time value is 41ns. For example, with a 100MHz clock, max. of 4 clock periods are required. So, the value of the register is 0x2.

TDS_TIMER_Reg

SDA data setup time during both Open-Drain/Push-Pull mode (in Number of Reference clock).

Bit #	R/W	Name	Reset	Description
2:0	R/W	TDS_TIMER	0x1	SDA Set up time (number of clock cycles). As per specification, min time value is 3ns. For example, with a 100MHz clock, 1 clock period is required. So, the value of the register is 0x1.

THD_PP_TIMER_Reg

SDA data hold time during the Push-Pull mode (in Number of Reference clock).

Bit #	R/W	Name	Reset	Description
2:0	R/W	THD_PP_TIMER	0x1	SDA Hold time (number of clock cycles). As per specification, min time value is 6ns. For example, with a 100MHz clock, 1 clock period is required. So, the value of the register is 0x1.

TCBP_TIMER_Reg

Clock time before Stop condition.

Bit #	R/W	Name	Reset	Description
4:0	R/W	TCBP_TIMER	0x1	To signal stop condition, the master should change the SDA pin after this time of SCL clock edge (number of clock cycles). As per specification, min time value is 19.2ns. For example, with a 100MHz ref clock, 2 clock periods are required. So, the value of the register is 0x1.

TCBSR_TIMER_Reg

Clock time before Repeated start condition.

Bit #	R/W	Name	Reset	Description
2:0	R/W	TCBSR_TIMER	0x1	To signal repeated start condition, the master should change the SDA pin after this time of SCL clock edge (number of clock cycles). As per specification, min time value is 19.2ns. For example, with a 100MHz ref clock, 2 clock periods are required. So, the value of the register is 0x1.

THD_DDR_TIMER_Reg

SDA data hold time during the Push-Pull mode (in Number of Reference clock) in DDR Data rate.

Bit #	R/W	Name	Reset	Description
2:0	R/W	THD_DDR_TIMER	0x1	SDA Hold time (number of clock cycles). As per specification, min time value is 6ns. For example, with a 100MHz ref clock, 1 clock period is required. So, the value of the register is 0x1.

BUS_FREE_TIMER_Reg

Bus free time between the Stop condition and the next start condition (in Number of Reference clock).

Bit #	R/W	Name	Reset	Description
31:0	R/W	BUS_FREE_TIMER	0xC8	Bus free time after the stop condition and the start condition. As per specification, min. value of this period is 1us.

BUS_AVAIL_TIMER_Reg

Time to keep the SDA and SCL pin to High (in Number of Reference clock).

Bit #	R/W	Name	Reset	Description
31:0	R/W	BUS_AVAIL_TIMER	0xC8	Bus available time to drive SCL and SDA pin to high. As per specification, min. value of this period is 1us.

TIDLE_TIMER_Reg

Extended duration of the bus free condition after the Stop condition (in Number of Reference clock) to enable the device to drive the Hot Join request.

Bit #	R/W	Name	Reset	Description
31:0	R/W	TIDLE_TIMER	0xC8	Bus available time to drive SCL and SDA pin to high. As per specification, min. value of this period is 1ms.

TSCO_TIMER_Reg

Maximum time the slave needs to drive the bus during the ACK/read data after the clock change.

Bit #	R/W	Name	Reset	Description
2:0	R/W	TSCO_TIMER	0x2	Maximum time for the slave to release bus after the clock change. As per specification, max. value is 12ns.

TSU_STA_TIMER_Reg

SDA data setup time during both Open-Drain (in Number of Reference clock) for a Repeated Start.

Bit #	R/W	Name	Reset	Description
8:0	R/W	TSU_STA_TIMER	0x46	SDA setup time (number of clock cycles) for a repeated start during legacy I2C mode. As per specification, min time value is 600ns for legacy FM mode, and 260ns for legacy FM+ mode. For example, with a 100MHz clock, 70 clock periods are required for FM mode, and the value of the register is 0x46. For FM+ mode, the value of this register is 0x1E.

THD_STA_TIMER_Reg

SDA data hold time during the Open Drain mode (in Number of Reference clock).

Bit #	R/W	Name	Reset	Description
8:0	R/W	THD_STA_TIMER	0x46	SDA hold time (number of clock cycles) after start/repeated start. As per specification, min time value is 600ns for legacy FM mode, and 260ns for legacy FM+ mode. For example, with a 100MHz clock, 70 clock periods are required for FM mode, and the value of the register is 0x46. For FM+ mode, the value of this register is 0x1E.

TLOW_TIMER_Reg

Low period of SCL pin in Open Drain mode during Legacy I2C Mode (in Number of Reference clock) after the timer reached the SCL pin can go high.

Bit #	R/W	Name	Reset	Description
15:0	R/W	TLOW_TIMER	0x8C	Low period of the SCL clock pin (number of clock cycles) in legacy I2C mode. As per specification, min time value is 1300ns for legacy FM mode, and 500ns for legacy FM+ mode. For example, with a 100MHz clock, 130 clock periods are required for FM mode, and the value of the register is 0x8C. For FM+ mode, the value of this register is 0x3C.

THIGH_TIMER_Reg

High period of SCL pin in Open Drain Mode for Legacy I2C (in Number of Reference clock) after this timer count reached the SCL pin can go low.

Bit #	R/W	Name	Reset	Description
15:0	R/W	THIGH_TIMER	0x46	High period of the SCL clock pin (number of clock cycles) in legacy I2C mode. As per specification, min time value is 600ns for legacy FM mode, and 260ns for legacy FM+ mode. For example, with a 100MHz clock, 70 clock periods are required for FM mode, and the value of the register is 0x46. For FM+ mode, the value of this register is 0x1E.

TVD_DATA_TIMER_Reg

Data hold time in Open Drain Mode for Legacy I2C (in Number of Reference clock) after this timer count reached the SDA pin can change its value.

Bit #	R/W	Name	Reset	Description
9:0	R/W	TVD_DATA_TIMER	0x46	Data hold time (number of clock cycles) in legacy I2C mode. As per specification, min data setup time value is 100ns for legacy FM mode (so hold time will be min 600ns), and 50ns for legacy FM+ mode (so hold time will be min 260ns). For example, with a 100MHz clock, 70 clock periods are required for FM mode, and the value of the register is 0x46. For FM+ Mode, the value of this register is 0x1E.

TSU_STOP_TIMER_Reg

SDA data setup time during Open-Drain (in Number of Reference clock) for Stop condition.

Bit #	R/W	Name	Reset	Description
8:0	R/W	TSU_STOP_TIMER	0x46	SDA setup time (number of clock cycles) for stop condition during legacy I2C mode. As per specification, min time value is 600ns for legacy FM mode, and 260ns for Legacy FM+ Mode. For example, with a 100MHz clock, 70 clock periods are required for FM mode, and the value of the register is 0x46. For FM+ Mode, the value of this register is 0x1E.

Device_Addr_Table_Reg0

Device type, IBI handling and Dynamic address to be set and Static address of the slave0.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg1

Device type, IBI handling and Dynamic address to be set and Static address of the slave1.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg2

Device type, IBI handling and Dynamic address to be set and Static address of the slave2.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg3

Device type, IBI handling and Dynamic address to be set and Static address of the slave3.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg4

Device type, IBI handling and Dynamic address to be set and Static address of the slave4.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg5

Device type, IBI handling and Dynamic address to be set and Static address of the slave5.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg6

Device type, IBI handling and Dynamic address to be set and Static address of the slave6.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg7

Device type, IBI handling and Dynamic address to be set and Static address of the slave7.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg8

Device type, IBI handling and Dynamic address to be set and Static address of the slave8.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg9

Device type, IBI handling and Dynamic address to be set and Static address of the slave9.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg10

Device type, IBI handling and Dynamic address to be set and Static address of the slave10.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Addr_Table_Reg11

Device type, IBI handling and Dynamic address to be set and Static address of the slave11.

Bit #	R/W	Name	Reset	Description
6:0	R/W	Static_Addr	0x0	Device I3C/I2C static address
12	R/W	IBI_Payload	0x0	IBI payload. This bit reflects the information from the BCR regarding the IBI data present to be received by the master: b0: No IBI data payload; b1: IBI data contains payload
13	R/W	In_Band_Req	0x0	In-band interrupt enable. Controls the master to ACK/NACK the IBI requests from the particular slave.
15	R/W	Timestamp	0x0	Device IBI timestamp. Enables or disables timestamping for a particular device: b0: Timestamp not required; b1: Timestamp enabled
23:16	R/W	Dynamic_Addr	0x0	Device I3C dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave
31	R/W	Device_Type	0x0	Device type: b0: I3C Device; b1: I2C Device

Device_Char_Table_Reg0_0

MSB of PID value of slave0 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_0

LSB of PID value of slave0 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_0

BCR, DCR and dynamic address of slave0 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_1

MSB of PID value of slave1 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_1

LSB of PID value of slave1 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_1

BCR, DCR and dynamic address of slave1 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_2

MSB of PID value of slave2 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_2

LSB of PID value of slave2 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_2

BCR, DCR and dynamic address of slave2 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_3

MSB of PID value of slave3 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_3

LSB of PID value of slave3 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_3

BCR, DCR and dynamic address of slave3 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_4

MSB of PID value of slave4 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_4

LSB of PID value of slave4 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_4

BCR, DCR and dynamic address of slave4 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_5

MSB of PID value of slave5 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_5

LSB of PID value of slave5 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_5

BCR, DCR and dynamic address of slave5 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_6

MSB of PID value of slave6 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_6

LSB of PID value of slave6 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_6

BCR, DCR and dynamic address of slave6 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_7

MSB of PID value of slave7 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_7

LSB of PID value of slave7 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_7

BCR, DCR and dynamic address of slave7 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_8

MSB of PID value of slave8 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_8

LSB of PID value of slave8 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_8

BCR, DCR and dynamic address of slave8 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_9

MSB of PID value of slave9 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_9

LSB of PID value of slave9 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_9

BCR, DCR and dynamic address of slave9 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_10

MSB of PID value of slave10 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_10

LSB of PID value of slave10 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_10

BCR, DCR and dynamic address of slave10 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

Device_Char_Table_Reg0_11

MSB of PID value of slave11 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
31:0	R	PID_HIGH	0x0	Device provisional ID high: bits [48:16] of Device's I3C PID.

Device_Char_Table_Reg1_11

LSB of PID value of slave11 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
15:0	R	PID_LOW	0x0	Device provisional ID low: bits [15:0] of Device's I3C PID.

Device_Char_Table_Reg2_11

BCR, DCR and dynamic address of slave11 during the dynamic address assignment command.

Bit #	R/W	Name	Reset	Description
7:0	R	DCR	0x0	Device Characteristics Register(DCR) of the slave.
15:8	R	BCR	0x0	I3C Bus Characteristics Register(BCR) of the slave.
23:16	R	Dynamic_Addr	0x0	Device I3C Dynamic address: bit 23 is the parity bit for the dynamic address calculated by software, bits 22:16 are the dynamic address of the slave

5.2.11.3 I3C Interface Commands

Command name	Width	Command code	Description
REGULAR_CMD	64	0x0	Regular command
IMM_TRANSFER	64	0x1	Immediate transfer command
ADDR_ASSIGN	64	0x2	Address assignment command

REGULAR_CMD

Bit #	Name	Description
63:48	DATA_LEN	Indicates the number of bytes to be transferred
31	TOC	Controls the bus condition when completing the data transfer: b0: Repeated start for next transfer; b1: Stop condition
30	ROC	Controls the response status after the completion of transfer: b0: Response status not required; b1: Response status required
29	RnW	Transfer direction: b0: Write transfer; b1: Read transfer
28:26	MODE	Mode and speed of the transfer using I3C or I2C bus. Values for I3C mode: h0: Standard speed mode (12.5MHZ); h1: 8MHz speed mode; h2: 6MHz speed mode; h3: 4MHz speed mode; h4: 2MHz speed mode; h5: Reserved; h6: HDR-DDR mode (25MHz speed); h7: Reserved. Values for I2C Mode: h0: I2C FM; h1: I2C FM+; h2-h7: Reserved.
21	BC_EN	Initiate the transfer after sending the Broadcast message (7'h7E): b0: No Broadcast message required; b1: Broadcast message required
20:16	DEV_INDEX	Slave address to which the data transfer is required
15	CP	Indicates if the command field is valid or not: b0: SDR transfer, no CCC command; b1: CCC or HDR command transfer
14:7	CMD	I3C command code: 8-bit for CCC, 7-bit for HDR
6:3	TID	Command request Tag ID. Maximum of 16 transfers can be initiated.
2:0	CMD_ATTR	Command code – 0x0 for regular command type

IMM_TRANSFER

Bit #	Name	Description
63:56	DATA_BYTE_4	Data byte 4 to be transferred.
55:48	DATA_BYTE_3	Data byte 3 to be transferred.
47:40	DATA_BYTE_2	Data byte 2 to be transferred.
39:32	DATA_BYTE_1	Data byte 1 to be transferred.
31	TOC	Controls the bus condition when completing the data transfer: b0: Repeated start for next transfer; b1: Stop condition
30	ROC	Controls the response status after the completion of transfer: b0: Response status not required; b1: Response status required
29	RnW	Transfer direction: b0: Write transfer; b1: Read transfer
28:26	MODE	Mode and speed of the transfer using I3C or I2C bus. Values for I3C mode: h0: Standard speed mode (12.5MHZ); h1: 8MHz speed mode; h2: 6MHz speed mode; h3: 4MHz speed mode; h4: 2MHz speed mode; h5: Reserved; h6: HDR-DDR mode (25MHz speed); h7: Reserved. Values for I2C Mode: h0: I2C FM; h1: I2C FM+; h2-h7: Reserved.
25:23	BYTE_CNT	Number of valid data bytes: h0: No payload; h1 to h4: 1 to 4 bytes respectively; h5 to h7: Reserved.
21	BC_EN	Initiate the transfer after sending the Broadcast message (7'h7E): b0: No Broadcast message required; b1: Broadcast message required
20:16	DEV_INDEX	Slave address to which the data transfer is required
15	CP	Indicates if the command field is valid or not: b0: SDR transfer, no CCC command; b1: CCC or HDR command transfer
14:7	CMD	I3C command code: 8-bit for CCC, 7-bit for HDR
6:3	TID	Command request Tag ID. Maximum of 16 transfers can be initiated.
2:0	CMD_ATTR	Command code – 0x1 for immediate transfer command type

ADDR_ASSIGN

Bit #	Name	Description
31	TOC	Controls the bus condition when completing the data transfer: b0: Repeated start for next transfer; b1: Stop condition
30	ROC	Controls the response status after the completion of transfer: b0: Response status not required; b1: Response status required
29:26	DEVICE_CNT	Device count. Master to perform the dynamic address assignment for the number of devices present.
21	BC_EN	Initiate the transfer after sending the Broadcast message (7'h7E): b0: No Broadcast message required; b1: Broadcast message required
20:16	DEV_INDEX	Slave address to which the data transfer is required
14:7	CMD	ENTDAA or SETDASA command code
6:3	TID	Command request Tag ID. Maximum of 16 transfers can be initiated.
2:0	CMD_ATTR	Command code – 0x2 for address assignment command type

5.2.12 Timers

GAP9 offers two timer units, each one manages the following features:

- Two general purpose 32-bit up counters per unit, able to work in 64-bit cascaded mode
- Input trigger sources:
  - FLL clock
  - FLL clock + Prescaler
  - Slow reference clock (REF SLOW clock or divided REF FAST clock, see the System Clocks section)
  - External event
- 8-bit programmable prescaler to FLL clock
- Counting modes:
  - One shot mode: timer is stopped after first comparison match
  - Continuous mode: timer continues counting after comparison match
  - Cycle mode: timer resets to 0 after comparison match and continues counting
- Interrupt request generation on comparison match

5.2.12.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_LO	0x0		Timer Low Configuration register
CFG_HI	0x4		Timer High Configuration register
CNT_LO	0x8		Timer Low counter value register
CNT_HI	0xC		Timer High counter value register
CMP_LO	0x10		Timer Low comparator value register
CMP_HI	0x14		Timer High comparator value register
START_LO	0x18		Start Timer Low counting register
START_HI	0x1C		Start Timer High counting register
RESET_LO	0x20		Reset Timer Low counter register
RESET_HI	0x24		Reset Timer High counter register

CFG_LO

Timer Low Configuration register

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	Timer Low enable: b0: disabled; b1: enabled
1	R/W	RESET	0x0	Write b1 to reset Timer Low counter. This fiels is cleared after reset execution
2	R/W	IRQEN	0x0	Timer Low compare interrupt enable: b0: disabled; b1: enabled
3	R/W	IEM	0x0	Timer Low input event mask: b0: disabled; b1: enabled
4	R/W	MODE	0x0	Timer Low continuous mode configuration: b0: Timer Low counter continues incrementing when value matches CMP_LO; b1: Timer low counter is reset when value matches CMP_LO
5	R/W	ONE_S	0x0	Timer Low one shot configuration: b0: Timer Low remains enabled when value matches CMP_LO; b1: disable Timer Low when value matches CMP_LO
6	R/W	PEN	0x0	Timer Low prescaler enable: b0: disabled; b1: enabled
7	R/W	CCFG	0x0	Timer Low clock source configuration: b0: FLL or FLL+Prescaler; b1: slow reference clock
15:8	R/W	PVAL	0x0	Timer Low prescaler value. Ftimer = Fclk / (1 + PVAL)
31	R/W	CASC	0x0	Timer Low + Timer High 64bit cascaded mode: b0: disabled; b1: enabled

CFG_HI

Timer High Configuration register

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	Timer High enable: b0: disabled; b1: enabled
1	W	RESET	0x0	Write b1 to reset Timer High counter. This fiels is cleared after reset execution
2	R/W	IRQEN	0x0	Timer High compare interrupt enable: b0: disabled; b1: enabled
3	R/W	IEM	0x0	Timer High input event mask: b0: disabled; b1: enabled
4	R/W	MODE	0x0	Timer High continuous mode configuration: b0: Timer Low counter continues incrementing when value matches CMP_HI; b1: Timer low counter is reset when value matches CMP_HI
5	R/W	ONE_S	0x0	Timer High one shot configuration: b0: Timer Low remains enabled when value matches CMP_HI; b1: disable Timer Low when value matches CMP_HI
6	R/W	PEN	0x0	Timer High prescaler enable: b0: disabled; b1: enabled
7	R/W	CLKCFG	0x0	Timer High clock source configuration: b0: FLL or FLL+Prescaler; b1: slow reference clock

CNT_LO

Timer Low counter value register

Bit #	R/W	Name	Reset	Description
31:0	R/W	CNT_LO	0x0	Timer Low counter value

CNT_HI

Timer High counter value register

Bit #	R/W	Name	Reset	Description
31:0	R/W	CNT_HI	0x0	Timer High counter value

CMP_LO

Timer Low comparator value register

Bit #	R/W	Name	Reset	Description
31:0	R/W	CMP_LO	0x0	Timer Low comparator value

CMP_HI

Timer High comparator value register

Bit #	R/W	Name	Reset	Description
31:0	R/W	CMP_HI	0x0	Timer High comparator value

START_LO

Start Timer Low counting register

Bit #	R/W	Name	Reset	Description
0	W	STRT_LO	0x0	Timer Low start command. When executed, CFG_LO ENABLE field is set

START_HI

Start Timer High counting register

Bit #	R/W	Name	Reset	Description
0	W	STRT_HI	0x0	Timer High start command. When executed, CFG_HI ENABLE field is set

RESET_LO

Reset Timer Low counter register

Bit #	R/W	Name	Reset	Description
0	W	RST_LO	0x0	Timer Low counter reset command. When executed, CFG_LO RESET field is set

RESET_HI

Reset Timer High counter register

Bit #	R/W	Name	Reset	Description
0	W	RST_HI	0x0	Timer High counter reset command. When executed, CFG_HI RESET field is set

5.2.13 Memory Protection Unit

The memory protection unit controls the different MPU filters available in GAP9.

5.2.13.1 MPU Behavior

The Memory Protection Unit offers a global enable/disable control, and configurable rules for each MPU filter. The following table lists the different MPU filters, and the number of available rules for each of them:

Origin of access	Filter name	Number of rules
FC data	FC_DATA	12
FC instructions	FC_INSTR	4
Cluster	CLUSTER	8
Debug (TAP)	DEBUG	12
uDMA	UDMA	4

GAP9 memory protection unit handles pages of 128 bytes, each page being identified by a 23-bit address. This means that the MPU covers the 0x00000000-0x3FFFFFFF memory range. A rule is defined as an address range (more precisely a 128-byte-page range) and an authorized access mode (read and/or write and/or execute).

When enabled, MPU filters are by default blocking all accesses made in user mode, i.e. every access except FC accesses made in machine mode. For an access in user mode to be granted, it must be allowed by at least one of the active rules.

Each rule of each MPU filter is configured using a dedicated pair of registers: MPU_<filter name>_RULE_START_<rule index> and MPU_<filter name>_RULE_END_<rule index>. The *_START_* register contains the address of the first 128-byte page for this rule, as well as a bit to enable or disable this rule (bit 23). The *_END_* register contains the address of the last 128-byte page for this rule, and the authorized operations (bits 25, 24 and 23 are used to control Execute, Write and Read access rights, respectively).

In case of a forbidden access, the MPU updates the corresponding filter's MPU_<filter name>_SREG and MPU_<filter name>_ERR_ADDR registers. Note that those registers will then keep their values – whatever new forbidden access is detected – until they are cleared by writing any value into them. The MPU_ERR_IRQ interrupt is raised for every forbidden access.

Note: FC accesses in machine mode are always granted.

To open access to a region, a program needs to first configure the *_END_* register with the 23-bit address of the *stop page* address and the access mode, then configure the *_START_* register with the 23-bit *start page* address and bit 23 set to 1 to enable access. To reconfigure a rule, bit 23 of *_START_* register must first be set to 0, in order to disable the previous setting.

For instance, this code enables read and write accesses from cluster to the 0x1C000000 to 0x1C00FFFF address range (execute access is not allowed):

```
/* END register address          stop page address   read access   write access   no execute access */
*MPU_CLUSTER_RULE_END_0_ADDR   = (0x1C00FFFF >> 7) |  (1 << 23)  |  (1 << 24)   |   (0 << 25);
/* START register address       start page address   enable rule */
*MPU_CLUSTER_RULE_START_0_ADDR = (0x1C000000 >> 7) |  (1 << 23);
```

5.2.13.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
MPU_ENABLE	0x0	32	Global enable for all MPU filters
MPU_FC_DATA_RULE_START_0	0x10	32	Start page for rule 0 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_0	0x14	32	End page for rule 0 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_1	0x18	32	Start page for rule 1 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_1	0x1C	32	End page for rule 1 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_2	0x20	32	Start page for rule 2 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_2	0x24	32	End page for rule 2 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_3	0x28	32	Start page for rule 3 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_3	0x2C	32	End page for rule 3 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_4	0x30	32	Start page for rule 4 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_4	0x34	32	End page for rule 4 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_5	0x38	32	Start page for rule 5 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_5	0x3C	32	End page for rule 5 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_6	0x40	32	Start page for rule 6 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_6	0x44	32	End page for rule 6 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_7	0x48	32	Start page for rule 7 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_7	0x4C	32	End page for rule 7 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_8	0x50	32	Start page for rule 8 of FC_DATA MPU filter

Name	Offset	Width	Description
MPU_FC_DATA_RULE_END_8	0x54	32	End page for rule 8 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_9	0x58	32	Start page for rule 9 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_9	0x5C	32	End page for rule 9 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_10	0x60	32	Start page for rule 10 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_10	0x64	32	End page for rule 10 of FC_DATA MPU filter
MPU_FC_DATA_RULE_START_11	0x68	32	Start page for rule 11 of FC_DATA MPU filter
MPU_FC_DATA_RULE_END_11	0x6C	32	End page for rule 11 of FC_DATA MPU filter
MPU_FC_INSTR_RULE_START_0	0x70	32	Start page for rule 0 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_END_0	0x74	32	End page for rule 0 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_START_1	0x78	32	Start page for rule 1 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_END_1	0x7C	32	End page for rule 1 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_START_2	0x80	32	Start page for rule 2 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_END_2	0x84	32	End page for rule 2 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_START_3	0x88	32	Start page for rule 3 of FC_INSTR MPU filter
MPU_FC_INSTR_RULE_END_3	0x8C	32	End page for rule 3 of FC_INSTR MPU filter
MPU_CLUSTER_RULE_START_0	0x90	32	Start page for rule 0 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_0	0x94	32	End page for rule 0 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_1	0x98	32	Start page for rule 1 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_1	0x9C	32	End page for rule 1 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_2	0xA0	32	Start page for rule 2 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_2	0xA4	32	End page for rule 2 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_3	0xA8	32	Start page for rule 3 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_3	0xAC	32	End page for rule 3 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_4	0xB0	32	Start page for rule 4 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_4	0xB4	32	End page for rule 4 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_5	0xB8	32	Start page for rule 5 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_5	0xBC	32	End page for rule 5 of CLUSTER MPU filter

Name	Offset	Width	Description
MPU_CLUSTER_RULE_START_6	0xC0	32	Start page for rule 6 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_6	0xC4	32	End page for rule 6 of CLUSTER MPU filter
MPU_CLUSTER_RULE_START_7	0xC8	32	Start page for rule 7 of CLUSTER MPU filter
MPU_CLUSTER_RULE_END_7	0xCC	32	End page for rule 7 of CLUSTER MPU filter
MPU_DEBUG_RULE_START_0	0xD0	32	Start page for rule 0 of DEBUG MPU filter
MPU_DEBUG_RULE_END_0	0xD4	32	End page for rule 0 of DEBUG MPU filter
MPU_DEBUG_RULE_START_1	0xD8	32	Start page for rule 1 of DEBUG MPU filter
MPU_DEBUG_RULE_END_1	0xDC	32	End page for rule 1 of DEBUG MPU filter
MPU_DEBUG_RULE_START_2	0xE0	32	Start page for rule 2 of DEBUG MPU filter
MPU_DEBUG_RULE_END_2	0xE4	32	End page for rule 2 of DEBUG MPU filter
MPU_DEBUG_RULE_START_3	0xE8	32	Start page for rule 3 of DEBUG MPU filter
MPU_DEBUG_RULE_END_3	0xEC	32	End page for rule 3 of DEBUG MPU filter
MPU_DEBUG_RULE_START_4	0xF0	32	Start page for rule 4 of DEBUG MPU filter
MPU_DEBUG_RULE_END_4	0xF4	32	End page for rule 4 of DEBUG MPU filter
MPU_DEBUG_RULE_START_5	0xF8	32	Start page for rule 5 of DEBUG MPU filter
MPU_DEBUG_RULE_END_5	0xFC	32	End page for rule 5 of DEBUG MPU filter
MPU_DEBUG_RULE_START_6	0x100	32	Start page for rule 6 of DEBUG MPU filter
MPU_DEBUG_RULE_END_6	0x104	32	End page for rule 6 of DEBUG MPU filter
MPU_DEBUG_RULE_START_7	0x108	32	Start page for rule 7 of DEBUG MPU filter
MPU_DEBUG_RULE_END_7	0x10C	32	End page for rule 7 of DEBUG MPU filter
MPU_DEBUG_RULE_START_8	0x110	32	Start page for rule 8 of DEBUG MPU filter
MPU_DEBUG_RULE_END_8	0x114	32	End page for rule 8 of DEBUG MPU filter
MPU_DEBUG_RULE_START_9	0x118	32	Start page for rule 9 of DEBUG MPU filter
MPU_DEBUG_RULE_END_9	0x11C	32	End page for rule 9 of DEBUG MPU filter
MPU_DEBUG_RULE_START_10	0x120	32	Start page for rule 10 of DEBUG MPU filter
MPU_DEBUG_RULE_END_10	0x124	32	End page for rule 10 of DEBUG MPU filter
MPU_DEBUG_RULE_START_11	0x128	32	Start page for rule 11 of DEBUG MPU filter

Name	Offset	Width	Description
MPU_DEBUG_RULE_END_11	0x12C	32	End page for rule 11 of DEBUG MPU filter
MPU_UDMA_RULE_START_0	0x130	32	Start page for rule 0 of UDMA MPU filter
MPU_UDMA_RULE_END_0	0x134	32	End page for rule 0 of UDMA MPU filter
MPU_UDMA_RULE_START_1	0x138	32	Start page for rule 1 of UDMA MPU filter
MPU_UDMA_RULE_END_1	0x13C	32	End page for rule 1 of UDMA MPU filter
MPU_UDMA_RULE_START_2	0x140	32	Start page for rule 1 of UDMA MPU filter
MPU_UDMA_RULE_END_2	0x144	32	End page for rule 2 of UDMA MPU filter
MPU_UDMA_RULE_START_3	0x148	32	Start page for rule 3 of UDMA MPU filter
MPU_UDMA_RULE_END_3	0x14C	32	End page for rule 3 of UDMA MPU filter
MPU_FC_DATA_SREG	0x150	32	FC_DATA MPU filter status register
MPU_FC_INSTR_SREG	0x154	32	FC_INSTR MPU filter status register
MPU_CLUSTER_SREG	0x158	32	CLUSTER filter status register
MPU_DEBUG_SREG	0x15C	32	DEBUG MPU filter status register
MPU_UDMA_SREG	0x160	32	UDMA MPU filter status register
MPU_FC_DATA_ERR_ADDR	0x164	32	FC_DATA bad address
MPU_FC_INSTR_ERR_ADDR	0x168	32	FC_INSTR bad address
MPU_CLUSTER_ERR_ADDR	0x16C	32	CLUSTER bad address
MPU_DEBUG_ERR_ADDR	0x170	32	DEBUG bad address
MPU_UDMA_ERR_ADDR	0x174	32	UDMA bad address

MPU_ENABLE

Global enable for all MPU filters

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	Global enable: when set to b1, all filter are enabled; when set to b0, all filters are in flow-through mode (no checks)

MPU_FC_DATA_RULE_START_0

Start page for rule 0 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_0

End page for rule 0 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_1

Start page for rule 1 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_1

End page for rule 1 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_2

Start page for rule 2 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_2

End page for rule 2 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_3

Start page for rule 3 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_3

End page for rule 3 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_4

Start page for rule 4 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_4

End page for rule 4 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_5

Start page for rule 5 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_5

End page for rule 5 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_6

Start page for rule 6 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_6

End page for rule 6 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_7

Start page for rule 7 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_7

End page for rule 7 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_8

Start page for rule 8 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_8

End page for rule 8 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_9

Start page for rule 9 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_9

End page for rule 9 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_10

Start page for rule 10 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_10

End page for rule 10 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_RULE_START_11

Start page for rule 11 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_DATA_RULE_END_11

End page for rule 11 of FC_DATA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_INSTR_RULE_START_0

Start page for rule 0 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_INSTR_RULE_END_0

End page for rule 0 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_INSTR_RULE_START_1

Start page for rule 1 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_INSTR_RULE_END_1

End page for rule 1 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_INSTR_RULE_START_2

Start page for rule 2 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_INSTR_RULE_END_2

End page for rule 2 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_INSTR_RULE_START_3

Start page for rule 3 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_FC_INSTR_RULE_END_3

End page for rule 3 of FC_INSTR MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_0

Start page for rule 0 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_0

End page for rule 0 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_1

Start page for rule 1 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_1

End page for rule 1 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_2

Start page for rule 2 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_2

End page for rule 2 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_3

Start page for rule 3 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_3

End page for rule 3 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_4

Start page for rule 4 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_4

End page for rule 4 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_5

Start page for rule 5 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_5

End page for rule 5 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_6

Start page for rule 6 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_6

End page for rule 6 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_CLUSTER_RULE_START_7

Start page for rule 7 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_CLUSTER_RULE_END_7

End page for rule 7 of CLUSTER MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_0

Start page for rule 0 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_0

End page for rule 0 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_1

Start page for rule 1 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_1

End page for rule 1 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_2

Start page for rule 2 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_2

End page for rule 2 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_3

Start page for rule 3 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_3

End page for rule 3 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_4

Start page for rule 4 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_4

End page for rule 4 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_5

Start page for rule 5 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_5

End page for rule 5 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_6

Start page for rule 6 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_6

End page for rule 6 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_7

Start page for rule 7 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_7

End page for rule 7 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_8

Start page for rule 8 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_8

End page for rule 8 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_9

Start page for rule 9 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_9

End page for rule 9 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_10

Start page for rule 10 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_10

End page for rule 10 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_DEBUG_RULE_START_11

Start page for rule 11 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_DEBUG_RULE_END_11

End page for rule 11 of DEBUG MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_UDMA_RULE_START_0

Start page for rule 0 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_UDMA_RULE_END_0

End page for rule 0 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_UDMA_RULE_START_1

Start page for rule 1 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_UDMA_RULE_END_1

End page for rule 1 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_UDMA_RULE_START_2

Start page for rule 1 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_UDMA_RULE_END_2

End page for rule 2 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_UDMA_RULE_START_3

Start page for rule 3 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	START_PAGE	0x00000	Page address of the first page of the address range subject to this filtering rule
23	R/W	STATUS	0x0	Rule enable: b1: enabled; b0: disabled

MPU_UDMA_RULE_END_3

End page for rule 3 of UDMA MPU filter

Bit #	R/W	Name	Reset	Description
22:0	R/W	END_PAGE	0x00000	Page address of the last page of the address range subject to this filtering rule
23	R/W	EN_READ	0x0	When set to b1, read operations are authorized by this rule
24	R/W	EN_WRITE	0x0	When set to b1, write operations are authorized by this rule
25	R/W	EN_EXEC	0x0	When set to b1, execute operations are authorized by this rule

MPU_FC_DATA_SREG

FC_DATA MPU filter status register

Bit #	R/W	Name	Reset	Description
11:0	R/W	RULE_MATCH	0x000	Gives all matching rules for the first encountered forbidden access; write any value to clear
14:12	R/W	FAILED_MODE	0x0	Gives the access mode failure (execute/write/read), if any, of the matching rule with the highest index, for the first encountered forbidden access – if a single rule matches, this is the failing access mode; write any value to clear

MPU_FC_INSTR_SREG

FC_INSTR MPU filter status register

Bit #	R/W	Name	Reset	Description
3:0	R/W	RULE_MATCH	0x0	Gives all matching rules for the first encountered forbidden access; write any value to clear
6:4	R/W	FAILED_MODE	0x0	Gives the access mode failure (execute/write/read), if any, of the matching rule with the highest index, for the first encountered forbidden access – if a single rule matches, this is the failing access mode; write any value to clear

MPU_CLUSTER_SREG

CLUSTER filter status register

Bit #	R/W	Name	Reset	Description
7:0	R/W	RULE_MATCH	0x00	Gives all matching rules for the first encountered forbidden access; write any value to clear
10:8	R/W	FAILED_MODE	0x0	Gives the access mode failure (execute/write/read), if any, of the matching rule with the highest index, for the first encountered forbidden access – if a single rule matches, this is the failing access mode; write any value to clear

MPU_DEBUG_SREG

DEBUG MPU filter status register

Bit #	R/W	Name	Reset	Description
11:0	R/W	RULE_MATCH	0x000	Gives all matching rules for the first encountered forbidden access; write any value to clear
14:12	R/W	FAILED_MODE	0x0	Gives the access mode failure (execute/write/read), if any, of the matching rule with the highest index, for the first encountered forbidden access – if a single rule matches, this is the failing access mode; write any value to clear

MPU_UDMA_SREG

UDMA MPU filter status register

Bit #	R/W	Name	Reset	Description
3:0	R/W	RULE_MATCH	0x0	Gives all matching rules for the first encountered forbidden access; write any value to clear
6:4	R/W	FAILED_MODE	0x0	Gives the access mode failure (execute/write/read), if any, of the matching rule with the highest index, for the first encountered forbidden access – if a single rule matches, this is the failing access mode; write any value to clear

MPU_FC_DATA_ERR_ADDR

FC_DATA bad address

Bit #	R/W	Name	Reset	Description
31:0	R/W	RULE_MATCH	0x00000000	Gives the address of the first encountered forbidden access; write any value to clear

MPU_FC_INSTR_ERR_ADDR

FC_INSTR bad address

Bit #	R/W	Name	Reset	Description
31:0	R/W	RULE_MATCH	0x00000000	Gives the address of the first encountered forbidden access; write any value to clear

MPU_CLUSTER_ERR_ADDR

CLUSTER bad address

Bit #	R/W	Name	Reset	Description
31:0	R/W	RULE_MATCH	0x00000000	Gives the address of the first encountered forbidden access; write any value to clear

MPU_DEBUG_ERR_ADDR

DEBUG bad address

Bit #	R/W	Name	Reset	Description
31:0	R/W	RULE_MATCH	0x00000000	Gives the address of the first encountered forbidden access; write any value to clear

MPU_UDMA_ERR_ADDR

UDMA bad address

Bit #	R/W	Name	Reset	Description
31:0	R/W	RULE_MATCH	0x00000000	Gives the address of the first encountered forbidden access; write any value to clear

5.2.14 EFuse

This peripheral manages the internal eFuses embedded in GAP9, and provides the following features:

- provides 128 one-time programmable 32-bit eFuses.
- access in read mode by byte.
- eFuse program bit per bit operation (each bit is 0 by default, can be programmed to 1, but cannot be set back to 0 afterwards).
- configurable low power sleep mode.

A number of eFuses are reserved to configure the behavior of the GAP9 boot from ROM. Other eFuses can be used for other purposes. The GAP9 ROM section shows the list of reserved eFuses.

5.2.14.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CMD	0x0	32	CMD register for eFuses
CFG	0x4	32	CFG register for timings
PAGE_PROTECT	0x8	32	Protection (RO mode) register for eFuse pages

CMD

CMD register for eFuses

Bit #	R/W	Name	Reset	Description
0	R/W	READ	0x0	Enable read mode
1	R/W	WRITE	0x0	Enable write mode
2	R/W	IDLE	0x0	Enable idle mode

CFG

CFG register for timings

Bit #	R/W	Name	Reset	Description
5:0	R/W	CNT_TARGET0	0x2	Counter value to generate short delays
15:6	R/W	CNT_TARGET1	0x32	Counter value to generate medium delays
29:16	R/W	CNT_TARGET2	0x1F4	Counter value to generate long delays
31:30	R/W	MARGIN	0x0	EFuse read margin

PAGE_PROTECT

Protection (RO mode) register for eFuse pages

Bit #	R/W	Name	Reset	Description
7:0	R/W	PROTECTED	0x0	Protected pages. Bit i is 1 means page i is in read only mode
31	R/W	LOCK	0x0	If 1, PAGE_PROTECT register cannot be overwritten

5.2.15 Quiddikey

5.2.15.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CR	0x0	32	Control register
SR	0x8	32	Status register
AR	0xC	32	Allow register
IER	0x10	32	Interrupt Enable register
IMR	0x14	32	Interrupt Mask register
ISR	0x18	32	Interrupt Status register
DATA_DEST	0x20	32	Destination Data register
DIR	0xA0	32	Data Input register
DOR	0xA8	32	Data Output register
MISC	0xC0	32	Miscellaneous register
IF_SR	0xD0	32	Interface Status register
TEST	0xD8	32	Test register
PSR	0xDC	32	PUF Score register
HW_RUC0	0xE0	32	Hardware Restrict User Context 0 register
HW_RUC1	0xE4	32	Hardware Restrict User Context 1 register
HW_SETTINGS	0xF0	32	Hardware Settings register
HW_INFO	0xF4	32	Hardware Information register
HW_ID	0xF8	32	Hardware Identifier register
HW_VER	0xFC	32	Hardware Version register

CR

Control register

Bit #	R/W	Name	Reset	Description
0	R/W	ZEROIZE	0x00	Begin Zeroize operation
1	R/W	ENROLL	0x00	Begin Enroll operation
2	R/W	START	0x00	Begin Start operation
5	R/W	STOP	0x00	Begin Stop operation
6	R/W	GET_KEY	0x00	Begin Get Key operation
7	R/W	UNWRAP	0x00	Begin Unwrap operation
8	R/W	WRAP_GENER-ATED_RANDOM	0x00	Begin Wrap Generation Random operation
9	R/W	WRAP	0x00	Begin Wrap operation
15	R/W	GENERATE_RANDOM	0x00	Begin Generate Random operation
16	R/W	RESEED	0x00	Begin Reseed operation
31	R/W	TEST_PUF	0x00	Begin Test PUF operation

SR

Status register

Bit #	R/W	Name	Reset	Description
0	R/W	BUSY	0x00	Operation is in progress
1	R/W	OK	0x00	Last operation was successful
2	R/W	ERROR	0x00	Last operation failed
3	R/W	ZEROIZED	0x00	Quiddikey is in Zeroized or in Locked state
4	R/W	REJECTED	0x00	Read: last command rejected, Write 1: Clear this bit
5	R/W	DI_REQUEST	0x00	Request for Data in transfer via DIR register
6	R/W	DO_REQUEST	0x00	Request for Data out transfer via DOR register
29	R/W	RESEED_WARNING	0x00	Reseed warning (see ORR RESEED_WARNING)
30	R/W	RESEED_REQUIRED	0x00	Reseed required (see ORR RESEED_REQUIRED)
31	R/W	LAB_TEST_MODE	0x00	Quiddikey is in state Lab Test Mode

AR

Allow register

Bit #	R/W	Name	Reset	Description
1	R/W	ALLOW_ENROLL	0x00	Operation allowed status
2	R/W	ALLOW_START	0x00	Operation allowed status
5	R/W	ALLOW_STOP	0x00	Operation allowed status
6	R/W	ALLOW_GET_KEY	0x00	Operation allowed status
7	R/W	ALLOW_UNWRAP	0x00	Operation allowed status
8	R/W	ALLOW_WRAP_-GEN_RND	0x00	Operation allowed status
9	R/W	ALLOW_WRAP	0x00	Operation allowed status
15	R/W	ALLOW_GEN_RND	0x00	Operation allowed status
16	R/W	ALLOW_RESEED	0x00	Operation allowed status
31	R/W	ALLOW_TEST_PUF	0x00	Operation allowed status

IER

Interrupt Enable register

Bit #	R/W	Name	Reset	Description
0	R/W	INT_EN	0x00	Interrupt enable register

IMR

Interrupt Mask register

Bit #	R/W	Name	Reset	Description
0	R/W	INT_EN_BUSY	0x00	Enable Busy interrupt
1	R/W	INT_EN_OK	0x00	Enable Ok interrupt
2	R/W	INT_EN_ERROR	0x00	Enable Error interrupt
3	R/W	INT_EN_ZEROIZED	0x00	Enable Zeroized interrupt
4	R/W	INT_EN_REJECTED	0x00	Enable Rejected interrupt
5	R/W	INT_EN_DI_REQUEST	0x00	Enable Data In Request interrupt
6	R/W	INT_EN_DO_REQUEST	0x00	Enable Data Out Request interrupt
29	R/W	INT_EN_RESEED_WARNING	0x00	Enable Reseed Warning interrupt
30	R/W	INT_EN_RESEED_REQUIRED	0x00	Enable Reseed Required interrupt
31	R/W	INT_EN_LAB_TEST_MODE	0x00	Enable Lab Test Mode interrupt

ISR

Interrupt Status register

Bit #	R/W	Name	Reset	Description
0	R/W	INT_BUSY	0x00	Busy interrupt status register
1	R/W	INT_OK	0x00	Ok interrupt status register
2	R/W	INT_ERROR	0x00	Error interrupt status register
3	R/W	INT_ZEROIZED	0x00	Zeroized interrupt status register
4	R/W	INT_REJECTED	0x00	Rejected interrupt status register
5	R/W	INT_DI_REQUEST	0x00	Data In Request interrupt status register
6	R/W	INT_DO_REQUEST	0x00	Data Out Request interrupt status register
29	R/W	INT_RESEED_WARNING	0x00	Reseed Warning interrupt status register
30	R/W	INT_RESEED_REQUIRED	0x00	Reseed Required interrupt status register
31	R/W	INT_LAB_TEST_MODE	0x00	Lab Test Mode interrupt status register

DATA_DEST

Destination Data register

Bit #	R/W	Name	Reset	Description
0	R/W	DEST_DOR	0x00	Data out register destination
1	R/W	DEST_SO	0x00	Secure Output interface destination

DIR

Data Input register

Bit #	R/W	Name	Reset	Description
31:0	R/W	DI	0x00	Data In field

DOR

Data Output register

Bit #	R/W	Name	Reset	Description
31:0	R	DOR	0x00	Data Out field

MISC

Miscellaneous register

Bit #	R/W	Name	Reset	Description
31:0	R	ENDIANNESS	0x00	Set endianness

IF_SR

Interface Status register

Bit #	R/W	Name	Reset	Description
0	R/W	APB_ERROR	0x00	An APB error has occured

TEST

Test register

Bit #	R/W	Name	Reset	Description
0	R/W	BIST_ENABLE	0x00	Isolates Quiddikey and runs BIST
4	R/W	BIST_RUNNING	0x00	BIST is in progress or finishing up
5	R/W	BIST_ACTIVE	0x00	BIST is in progress
6	R/W	BIST_OK	0x00	BIST has passed
7	R/W	BIST_ERROR	0x00	BIST has failed
31	R/W	ALLOW_BIST	0x00	BIST is not allowed

PSR

PUF Score register

Bit #	R/W	Name	Reset	Description
3:0	R/W	PUF_SCORE	0x00	PUF Score field

HW_RUC0

Hardware Restrict User Context 0 register

Bit #	R/W	Name	Reset	Description
31:0	R/W	RESTRICT_USER_-CONTEXT_0	0x00	Restrict User Context 0 field

HW_RUC1

Hardware Restrict User Context 1 register

Bit #	R/W	Name	Reset	Description
31:0	R/W	RESTRICT_USER_-CONTEXT_1	0x00	Restrict User Context 1 field

HW_SETTINGS

Hardware Settings register

Bit #	R/W	Name	Reset	Description
1	R/W	DISABLE_ENROLL	0x00	Enroll settings field
2	R/W	DISABLE_START	0x00	Start settings field
5	R/W	DISABLE_STOP	0x00	Stop settings field
6	R/W	DISABLE_GET_KEY	0x00	Get Key settings field
7	R/W	DISABLE_UNWRAP	0x00	Unwrap settings field
8	R/W	DISABLE_WRAP_-GEN_RND	0x00	Wrap Generated Random settings field
9	R/W	DISABLE_WRAP	0x00	Wrap settings field
15	R/W	DISABLE_GEN_RND	0x00	Generate Random settings field
16	R/W	DISABLE_RESEED	0x00	Reseed settings field
24	R/W	DISABLE_LAB_-TEST_MODE	0x00	Lab Test Mode settings field
25	R/W	SELECT_LAB_-TEST_MODE	0x00	Lab Test Mode select field
27	R/W	REQUIRE_RESEED_-SRC_VIA_DIR	0x00	Reseed via DIR settings field
28	R/W	REQUIRE_RESEED_-SRC_VIA_SI	0x00	Reseed via SI settings field
31	R/W	DISABLE_TEST_PUF	0x00	Test PUF settings field

HW_INFO

Hardware Information register

Bit #	R/W	Name	Reset	Description
21	R/W	CONFIG_SP_800_90	0x00	1: SP 800-90 is included, 0: not included
22	R/W	CONFIG_BIST	0x00	1: BIST is included, 0: not included
23	R/W	RESERVED	0x00	1: Safe, 0: Plus
24	R/W	CONFIG_WRAP	0x00	1: Wrap is included, 0: not included
31:28	R/W	CONFIG_TYPE	0x00	Quiddikey configuration

HW_ID

Hardware Identifier register

Bit #	R/W	Name	Reset	Description
31:0	R/W	HW_ID	0x00	Hardware Identifier

HW_VER

Hardware Version register

Bit #	R/W	Name	Reset	Description
7:0	R/W	HW_VER_REV	0x00	Hardware version, revision part
15:8	R/W	HW_VER_MINOR	0x00	Hardware version, minor part
23:16	R/W	HW_VER_MAJOR	0x00	Hardware version, major part

5.2.16 XIP

The XIP block (for eXecute In-Place) allows to execute code or fetch data from any UDMA flash/RAM. Instructions and data are then mapped in a virtual memory area starting @ 0x20000000 and ending @ 0x2FFFFFFF.

5.2.16.1 Details

Page size are defined in powers of 512 bytes. Maximum size is 64kB. If TLB mode is enabled, CFG_PAGE_SIZE0 is used for all pages/devices.

When NOT in TLB mode, CFG_SOFT_RESET, CFG_MNT_SIZE and CFG_PAGE_SIZE registers are used to create the virtual address boundaries for a peripheral to translate. CFG_EXT_ADDR register gives the matching physical address for the start of the virtual area, used for refills. For example if start of virtual address is set to 0x2100_0000, mount size to 0x1000 and page size to 0 (512 byte page size) then the virtual address area is from 0x2100_0000 to 0x2120_0000 = 0x2100_0000 + (0x1000 << 9). The virtual address has to hit inside this boundary.

If TLB mode is enabled, CFG_TLB_VIRT_PAGE and CFG_TLB_PHYS_PAGE are used to manage hits and misses:

- If one of the table entries matches the virtual address, then we take that entry is used to access the CFG_-PAGE(0-15). For example, if an access hits CFG_TLB_VIRT_PAGE_3, then CFG_PAGE3 is used to access the L2 memory
- When a miss occurs an address is assembled using the MSB bits from the TLB virtual address and the LSB bits from the physical one depending on the page size.

There are a number of extra constraints to comply with when using the XIP:

- CFG_TLB_VIRT_PAGE field must be written last at init (it gates the function of the IP)
- Both external memory addresses, virtual pages addresses and L2 cache pages must start aligned on corresponding device page size.
- Only Hyper_octo_SDIO0, Hyper_octo_SDIO1 and MRAM can be used as source for XIP.
- Live reconfiguration is not supported if executing or fetching data from XIP area. To reconfigure, use a dedicated code+data area in L2 to ensure atomicity. Also, flush icache.
- Address space maximum coverage is equal to 4K × PAGE_SIZE. This means that you will need 64kB pages to cover the whole address space.

5.2.16.2 Usage

Hyper/octo side:

- First open hyper or mram IP as usual & configure burst and such
- Set 'xip_en' bit in configuration

XIP side:

- Set configurations for chosen device (0: hyper0, 1: hyper1, 2: MRAM)
- First set virtual address start (between 0x20000000 and 0x2FFFFFFF)
- Then base external address (phys address in device)
- Set size (in pages)
- Page size: between 512B and 64kB
- then allocate L2 pages and set offset in CFG_PAGE* registers.

All addresses need to be aligned on chosen page size.

5.2.16.3 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_XIP	0x0	32	Main config register for XIP (TLB mode)
CFG_SOFT_RESET	0x4	32	Flush register for pages
CLUSTER_ERR_PAGE	0x8	32	Page address for first detected cluster error
ERR_PAGE	0xC	32	Page address for first detected FC instruction error
CFG_VIRT_ADDR0	0x10	32	Base virtual address for memory interface 0
CFG_VIRT_ADDR1	0x14	32	Base virtual address for memory interface 1
CFG_VIRT_ADDR2	0x18	32	Base virtual address for MRAM
CFG_EXT_ADDR0	0x1C	32	Base external address for memory interface 0
CFG_EXT_ADDR1	0x20	32	Base external address for memory interface 1
CFG_EXT_ADDR2	0x24	32	Base external address for MRAM
CFG_MNT_SIZE0	0x28	32	Virtual memory size for memory interface 0 (pages)
CFG_MNT_SIZE1	0x2C	32	Virtual memory size for memory interface 1 (pages)
CFG_MNT_SIZE2	0x30	32	Virtual memory size for MRAM (pages)
CFG_PAGE_SIZE0	0x34	32	Page size for memory interface 0 (512B–64kB, power of two)
CFG_PAGE_SIZE1	0x38	32	Page size for memory interface 1 (512B–64kB, power of two)
CFG_PAGE_SIZE2	0x3C	32	Page size for MRAM (512B–64kB, power of two)
CFG_PAGE0	0x40	32	Page 0 configuration
CFG_PAGE1	0x44	32	Page 1 configuration
CFG_PAGE2	0x48	32	Page 2 configuration
CFG_PAGE3	0x4C	32	Page 3 configuration
CFG_PAGE4	0x50	32	Page 4 configuration
CFG_PAGE5	0x54	32	Page 5 configuration

Name	Offset	Width	Description
CFG_PAGE6	0x58	32	Page 6 configuration
CFG_PAGE7	0x5C	32	Page 7 configuration
CFG_PAGE8	0x60	32	Page 8 configuration
CFG_PAGE9	0x64	32	Page 9 configuration
CFG_PAGE10	0x68	32	Page 10 configuration
CFG_PAGE11	0x6C	32	Page 11 configuration
CFG_PAGE12	0x70	32	Page 12 configuration
CFG_PAGE13	0x74	32	Page 13 configuration
CFG_PAGE14	0x78	32	Page 14 configuration
CFG_PAGE15	0x7C	32	Page 15 configuration
CFG_TLB_VIRT_PAGE_0	0x80	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_1	0x84	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_2	0x88	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_3	0x8C	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_4	0x90	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_5	0x94	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_6	0x98	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_7	0x9C	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_8	0xA0	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_9	0xA4	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_10	0xA8	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_11	0xAC	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_12	0xB0	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_13	0xB4	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_14	0xB8	32	TLB virtual page configuration
CFG_TLB_VIRT_PAGE_15	0xBC	32	TLB virtual page configuration
CFG_TLB_PHYS_PAGE_0	0xC0	32	TLB physical page configuration
CFG_TLB_PHYS_PAGE_1	0xC4	32	TLB physical page configuration
CFG_TLB_PHYS_PAGE_2	0xC8	32	TLB physical page configuration
CFG_TLB_PHYS_PAGE_3	0xCC	32	TLB physical page configuration
CFG_TLB_PHYS_PAGE_4	0xD0	32	TLB physical page configuration
CFG_TLB_PHYS_PAGE_5	0xD4	32	TLB physical page configuration

Name	Offset	Width	Description
CFG_TLB_PHYS_-PAGE_6	0xD8	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_7	0xDC	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_8	0xE0	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_9	0xE4	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_10	0xE8	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_11	0xEC	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_12	0xF0	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_13	0xF4	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_14	0xF8	32	TLB physical page configuration
CFG_TLB_PHYS_-PAGE_15	0xFC	32	TLB physical page configuration
CFG_XIP_LRU	0x100	32	TLB current LRU entry

CFG_XIP

Main config register for XIP (TLB mode)

Bit #	R/W	Name	Reset	Description
0	R/W	TLB_EN	0x0	Enable or Disable TLB mode
3:1	R/W	DEVICE_RO	0x0	Flag to check whether device is RO
16	R/W	POWER_ON	0x0	Inform XIP that cluster is powered on

CFG_SOFT_RESET

Flush register for pages

Bit #	R/W	Name	Reset	Description
0	R/W	RESET	0x0	Flush dirty pages, and reset logic to enable reconfiguration. Falls to 0 when done.

CLUSTER_ERR_PAGE

Page address for first detected cluster error

Bit #	R/W	Name	Reset	Description
31:0	R/W	VIRT_ADDR	0x20000000	Virtual address (must be in the [0x2000_0000-0x2FFF_FFFF] range)

ERR_PAGE

Page address for first detected FC instruction error

Bit #	R/W	Name	Reset	Description
31:0	R/W	VIRT_ADDR	0x20000000	Virtual address (must be in the [0x2000_0000-0x2FFF_FFFF] range)

CFG_VIRT_ADDR0

Base virtual address for memory interface 0

Bit #	R/W	Name	Reset	Description
31:0	R/W	VIRT_ADDR	0x20000000	Start of virtual address for external peripheral (must be in the [0x2000_0000-0x2FFF_FFFF] range)

CFG_VIRT_ADDR1

Base virtual address for memory interface 1

Bit #	R/W	Name	Reset	Description
31:0	R/W	VIRT_ADDR	0x20000000	Start of virtual address for external peripheral (must be in the [0x2000_0000-0x2FFF_FFFF] range)

CFG_VIRT_ADDR2

Base virtual address for MRAM

Bit #	R/W	Name	Reset	Description
31:0	R/W	VIRT_ADDR	0x20000000	Start of virtual address for external peripheral (must be in the [0x2000_0000-0x2FFF_FFFF] range)

CFG_EXT_ADDR0

Base external address for memory interface 0

Bit #	R/W	Name	Reset	Description
31:0	R/W	EXT_ADDR	0x0	Start of external address space for external peripheral

CFG_EXT_ADDR1

Base external address for memory interface 1

Bit #	R/W	Name	Reset	Description
31:0	R/W	EXT_ADDR	0x0	Start of external address space for external peripheral

CFG_EXT_ADDR2

Base external address for MRAM

Bit #	R/W	Name	Reset	Description
31:0	R/W	EXT_ADDR	0x0	Start of external address space for external peripheral

CFG_MNT_SIZE0

Virtual memory size for memory interface 0 (pages)

Bit #	R/W	Name	Reset	Description
15:0	R/W	MNT_SIZE	0x0	Size of the mounted region for external peripheral (in number of pages)

CFG_MNT_SIZE1

Virtual memory size for memory interface 1 (pages)

Bit #	R/W	Name	Reset	Description
15:0	R/W	MNT_SIZE	0x0	Size of the mounted region for external peripheral (in number of pages)

CFG_MNT_SIZE2

Virtual memory size for MRAM (pages)

Bit #	R/W	Name	Reset	Description
15:0	R/W	MNT_SIZE	0x0	Size of the mounted region for external peripheral (in number of pages)

CFG_PAGE_SIZE0

Page size for memory interface 0 (512B–64kB, power of two)

Bit #	R/W	Name	Reset	Description
2:0	R/W	PAGE_SIZE	0x0	Size of pages for external peripheral: 0: 512 Bytes; 1: 1 kBytes; 2: 2 kBytes; 3: 4 kBytes; ...; 7: 64 kBytes

CFG_PAGE_SIZE1

Page size for memory interface 1 (512B–64kB, power of two)

Bit #	R/W	Name	Reset	Description
2:0	R/W	PAGE_SIZE	0x0	Size of pages for external peripheral: 0: 512 Bytes; 1: 1 kBytes; 2: 2 kBytes; 3: 4 kBytes; ...; 7: 64 kBytes

CFG_PAGE_SIZE2

Page size for MRAM (512B–64kB, power of two)

Bit #	R/W	Name	Reset	Description
2:0	R/W	PAGE_SIZE	0x0	Size of pages for external peripheral: 0: 512 Bytes; 1: 1 kBytes; 2: 2 kBytes; 3: 4 kBytes; ...; 7: 64 kBytes

CFG_PAGE0

Page 0 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE1

Page 1 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE2

Page 2 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE3

Page 3 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE4

Page 4 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE5

Page 5 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE6

Page 6 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE7

Page 7 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE8

Page 8 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE9

Page 9 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE10

Page 10 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE11

Page 11 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE12

Page 12 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE13

Page 13 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE14

Page 14 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_PAGE15

Page 15 configuration

Bit #	R/W	Name	Reset	Description
20:0	R/W	INT_ADDR	0x0	21 LSB of L2 address of the page
28	R/W	CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_TLB_VIRT_PAGE
29	R/W	ACTIVE	0x0	Activate page: b0: page is ignored; b1: page is active
31:30	R/W	PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_TLB_VIRT_PAGE

CFG_TLB_VIRT_PAGE_0

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_1

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_2

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_3

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_4

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_5

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_6

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_7

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_8

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_9

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_10

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_11

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_12

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_13

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_14

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_VIRT_PAGE_15

TLB virtual page configuration

Bit #	R/W	Name	Reset	Description
27:9	R/W	TLB_VIRT_ADDR	0x0	Virtual address tag entry that is compared against the virtual address seeking for access
7	R	TLB_VALID_DATA	0x0	When system boots the data are invalid; when any register is written the data become valid
3:2	R/W	TLB_PER_ID	0x0	Peripheral ID: b00: memory interface 0; b01: memory interface 1; b10: MRAM; b11: reserved (field shared with corresponding CFG_PAGE
0	R/W	TLB_PAGE_-CACHEABLE	0x0	Make I-cache aware (b1) of this page or not (b0) (field shared with corresponding CFG_PAGE

CFG_TLB_PHYS_PAGE_0

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_1

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_2

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_3

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_4

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_5

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_6

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_7

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_8

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_9

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_10

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_11

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_12

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_13

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_14

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_TLB_PHYS_PAGE_15

TLB physical page configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	TLB_PHYS_ADDR	0x0	Physical address used along the virtual address in TLB to form the refill address in case of miss

CFG_XIP_LRU

TLB current LRU entry

Bit #	R/W	Name	Reset	Description
3:0	R	RESET	0x0	Least recently used entry

5.3 Micro DMA

5.3.1 UDMA Control

5.3.1.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_CG	0x0	32	uDMA peripherals clock gate configuration. This controls the individual clock-gating of each uDMA peripheral. There is one bit per peripheral, and the clock-gating is active low, i.e the peripheral is inactive when its corresponding bit is 0.
CFG_CG_SET	0x4	32	Each bit set to 1 sets the corresponding bit in the CFG_CG register
CFG_CG_CLR	0x8	32	Each bit set to 1 clears the corresponding bit in the CFG_CG register
CFG_RSTN	0xC	32	uDMA peripherals reset configuration (active low). At chip reset all periphs are under reset.
CFG_RSTN_SET	0x10	32	Each bit set to 1 sets the corresponding bit in the CFG_RSTN register
CFG_RSTN_CLR	0x14	32	Each bit set to 1 sets the corresponding bit in the CFG_RSTN register
CFG_EVENT	0x18	32	uDMA peripherals external event configuration
FIFO_CFG	0x1C	32	Set FIFO ID for push and pop

Name	Offset	Width	Description
FIFO_PUSHPOP_8	0x20	32	Push (write) and pop (read) 8-bit word to/from the FIFO
FIFO_PUSHPOP_16	0x24	32	Push (write) and pop (read) 16-bit word to/from the FIFO
FIFO_PUSHPOP_24	0x28	32	Push (write) and pop (read) 24-bit word to/from the FIFO
FIFO_PUSHPOP_32	0x2C	32	Push (write) and pop (read) 32-bit word to/from the FIFO
DATAMOVE_CFG	0x30	32	Configure data movement
DATAMOVE0_SIZE	0x34	32	Control of mover channel 0
DATAMOVE1_SIZE	0x38	32	Control of mover channel 1
STREAM_CFG	0x3C	32	Configure blocking behavior of streams
TIMEOUT_PRE0	0x40	32	Configuration of the frequency prescaler for timeout ch0
TIMEOUT_CH0	0x44	32	Configuration for timeout ch0
TIMEOUT_PRE1	0x48	32	Configuration of the frequency prescaler for timeout ch1
TIMEOUT_CH1	0x4C	32	Configuration for timeout ch1
TIMEOUT_PRE2	0x50	32	Configuration of the frequency prescaler for timeout ch2
TIMEOUT_CH2	0x54	32	Configuration for timeout ch2
TIMEOUT_PRE3	0x58	32	Configuration of the frequency prescaler for timeout ch3
TIMEOUT_CH3	0x5C	32	Configuration for timeout ch3
TIMEOUT_PRE4	0x60	32	Configuration of the frequency prescaler for timeout ch4
TIMEOUT_CH4	0x64	32	Configuration for timeout ch4
TIMEOUT_PRE5	0x68	32	Configuration of the frequency prescaler for timeout ch5
TIMEOUT_CH5	0x6C	32	Configuration for timeout ch5
TIMEOUT_PRE6	0x70	32	Configuration of the frequency prescaler for timeout ch6
TIMEOUT_CH6	0x74	32	Configuration for timeout ch6
TIMEOUT_PRE7	0x78	32	Configuration of the frequency prescaler for timeout ch7
TIMEOUT_CH7	0x7C	32	Configuration for timeout ch7

CFG_CG

uDMA peripherals clock gate configuration. This controls the individual clock-gating of each uDMA peripheral. There is one bit per peripheral, and the clock-gating is active low, i.e the peripheral is inactive when its corresponding bit is 0.

Bit #	R/W	Name	Reset	Description
0	R/W	CK_GATE_SPI0	0x0	Control internal clock of SPI0: 0: clock gated, 1: clock enabled
1	R/W	CK_GATE_SPI1	0x0	Control internal clock of SPI1: 0: clock gated, 1: clock enabled
2	R/W	CK_GATE_SPI2	0x0	Control internal clock of SPI2: 0: clock gated, 1: clock enabled
3	R/W	CK_GATE_SPI3	0x0	Control internal clock of SPI3: 0: clock gated, 1: clock enabled
4	R/W	CK_GATE_UART0	0x0	Control internal clock of UART0: 0: clock gated, 1: clock enabled
5	R/W	CK_GATE_UART1	0x0	Control internal clock of UART1: 0: clock gated, 1: clock enabled
6	R/W	CK_GATE_UART2	0x0	Control internal clock of UART2: 0: clock gated, 1: clock enabled
7	R/W	CK_GATE_UART3	0x0	Control internal clock of UART3: 0: clock gated, 1: clock enabled
8	R/W	CK_GATE_UART4	0x0	Control internal clock of UART4: 0: clock gated, 1: clock enabled
9	R/W	CK_GATE_I2C0	0x0	Control internal clock of I2C0: 0: clock gated, 1: clock enabled

Bit #	R/W	Name	Reset	Description
10	R/W	CK_GATE_I2C1	0x0	Control internal clock of I2C1: 0: clock gated, 1: clock enabled
11	R/W	CK_GATE_I2C2	0x0	Control internal clock of I2C2: 0: clock gated, 1: clock enabled
12	R/W	CK_GATE_I2C3	0x0	Control internal clock of I2C3: 0: clock gated, 1: clock enabled
13	R/W	CK_GATE_HYPER0	0x0	Control internal clock of memory interface 0: 0: clock gated, 1: clock enabled
14	R/W	CK_GATE_HYPER1	0x0	Control internal clock of memory interface 1: 0: clock gated, 1: clock enabled
15	R/W	CK_GATE_JTAG	0x0	Control internal clock of JTAG data transfers: 0: clock gated, 1: clock enabled
16	R/W	CK_GATE_SAI0	0x0	Control internal clock of I2S/SAI0: 0: clock gated, 1: clock enabled
17	R/W	CK_GATE_SAI1	0x0	Control internal clock of I2S/SAI1: 0: clock gated, 1: clock enabled
18	R/W	CK_GATE_SAI2	0x0	Control internal clock of I2S/SAI2: 0: clock gated, 1: clock enabled
19	R/W	CK_GATE_CPI	0x0	Control internal clock of camera parallel interface: 0: clock gated, 1: clock enabled
20	R/W	CK_GATE_CSI2	0x0	Control internal clock of CSI-2 interface: 0: clock gated, 1: clock enabled
21	R/W	CK_GATE_MRAM	0x0	Control internal clock of MRAM interface: 0: clock gated, 1: clock enabled
22	R/W	CK_GATE_FILTER	0x0	Control internal clock of filter unit: 0: clock gated, 1: clock enabled
23	R/W	CK_GATE_TIMES-TAMP	0x0	Control internal clock of timestamp unit: 0: clock gated, 1: clock enabled
24	R/W	CK_GATE_AES0	0x0	Control internal clock of AES0: 0: clock gated, 1: clock enabled
25	R/W	CK_GATE_AES1	0x0	Control internal clock of AES1: 0: clock gated, 1: clock enabled
26	R/W	CK_GATE_SFU	0x0	Control internal clock of SFU interface: 0: clock gated, 1: clock enabled
27	R/W	CK_GATE_FFC0	0x0	Control internal clock of fixed/floating point converter 0: 0: clock gated, 1: clock enabled
28	R/W	CK_GATE_FFC1	0x0	Control internal clock of fixed/floating point converter 1: 0: clock gated, 1: clock enabled
29	R/W	CK_GATE_FFC2	0x0	Control internal clock of fixed/floating point converter 2: 0: clock gated, 1: clock enabled
30	R/W	CK_GATE_FFC3	0x0	Control internal clock of fixed/floating point converter 3: 0: clock gated, 1: clock enabled

CFG_CG_SET

Each bit set to 1 sets the corresponding bit in the CFG_CG register

Bit #	R/W	Name	Reset	Description
0	W	CK_GATE_SPI0_SET	0x0	Write 1 to enable clock of SPI0
1	W	CK_GATE_SPI1_SET	0x0	Write 1 to enable clock of SPI1
2	W	CK_GATE_SPI2_SET	0x0	Write 1 to enable clock of SPI2
3	W	CK_GATE_SPI3_SET	0x0	Write 1 to enable clock of SPI3

Bit #	R/W	Name	Reset	Description
4	W	CK_GATE_UART0_-SET	0x0	Write 1 to enable clock of UART0
5	W	CK_GATE_UART1_-SET	0x0	Write 1 to enable clock of UART1
6	W	CK_GATE_UART2_-SET	0x0	Write 1 to enable clock of UART2
7	W	CK_GATE_UART3_-SET	0x0	Write 1 to enable clock of UART3
8	W	CK_GATE_UART4_-SET	0x0	Write 1 to enable clock of UART4
9	W	CK_GATE_I2C0_SET	0x0	Write 1 to enable clock of I2C0
10	W	CK_GATE_I2C1_SET	0x0	Write 1 to enable clock of I2C1
11	W	CK_GATE_I2C2_SET	0x0	Write 1 to enable clock of I2C2
12	W	CK_GATE_I2C3_SET	0x0	Write 1 to enable clock of I2C3
13	W	CK_GATE_HYPER0_-SET	0x0	Write 1 to enable clock of memory interface 0
14	W	CK_GATE_HYPER1_-SET	0x0	Write 1 to enable clock of memory interface 1
15	W	CK_GATE_JTAG_SET	0x0	Write 1 to enable clock of JTAG data transfers
16	W	CK_GATE_SAI0_SET	0x0	Write 1 to enable clock of I2S/SAI0
17	W	CK_GATE_SAI1_SET	0x0	Write 1 to enable clock of I2S/SAI1
18	W	CK_GATE_SAI2_SET	0x0	Write 1 to enable clock of I2S/SAI2
19	W	CK_GATE_CPI_SET	0x0	Write 1 to enable clock of camera parallel interface
20	W	CK_GATE_CSI2_SET	0x0	Write 1 to enable clock of CSI-2 interface
21	W	CK_GATE_MRAM_-SET	0x0	Write 1 to enable clock of MRAM interface
22	W	CK_GATE_FILTER_-SET	0x0	Write 1 to enable clock of filter unit
23	W	CK_GATE_TIMES-TAMP_SET	0x0	Write 1 to enable clock of timestamp unit
24	W	CK_GATE_AES0_SET	0x0	Write 1 to enable clock of AES0
25	W	CK_GATE_AES1_SET	0x0	Write 1 to enable clock of AES1
26	W	CK_GATE_SFU_SET	0x0	Write 1 to enable clock of SFU interface
27	W	CK_GATE_FFC0_SET	0x0	Write 1 to enable clock of fixed/floating point converter 0
28	W	CK_GATE_FFC1_SET	0x0	Write 1 to enable clock of fixed/floating point converter 1
29	W	CK_GATE_FFC2_SET	0x0	Write 1 to enable clock of fixed/floating point converter 2
30	W	CK_GATE_FFC3_SET	0x0	Write 1 to enable clock of fixed/floating point converter 3

CFG_CG_CLR

Each bit set to 1 clears the corresponding bit in the CFG_CG register

Bit #	R/W	Name	Reset	Description
0	W	CK_GATE_SPI0_CLR	0x0	Write 1 to gate clock of SPI0
1	W	CK_GATE_SPI1_CLR	0x0	Write 1 to gate clock of SPI1
2	W	CK_GATE_SPI2_CLR	0x0	Write 1 to gate clock of SPI2
3	W	CK_GATE_SPI3_CLR	0x0	Write 1 to gate clock of SPI3
4	W	CK_GATE_UART0_-CLR	0x0	Write 1 to gate clock of UART0

Bit #	R/W	Name	Reset	Description
5	W	CK_GATE_UART1_CLR	0x0	Write 1 to gate clock of UART1
6	W	CK_GATE_UART2_CLR	0x0	Write 1 to gate clock of UART2
7	W	CK_GATE_UART3_CLR	0x0	Write 1 to gate clock of UART3
8	W	CK_GATE_UART4_CLR	0x0	Write 1 to gate clock of UART4
9	W	CK_GATE_I2C0_CLR	0x0	Write 1 to gate clock of I2C0
10	W	CK_GATE_I2C1_CLR	0x0	Write 1 to gate clock of I2C1
11	W	CK_GATE_I2C2_CLR	0x0	Write 1 to gate clock of I2C2
12	W	CK_GATE_I2C3_CLR	0x0	Write 1 to gate clock of I2C3
13	W	CK_GATE_HYPER0_CLR	0x0	Write 1 to gate clock of memory interface 0
14	W	CK_GATE_HYPER1_CLR	0x0	Write 1 to gate clock of memory interface 1
15	W	CK_GATE_JTAG_CLR	0x0	Write 1 to gate clock of JTAG data transfers
16	W	CK_GATE_SAI0_CLR	0x0	Write 1 to gate clock of I2S/SAI0
17	W	CK_GATE_SAI1_CLR	0x0	Write 1 to gate clock of I2S/SAI1
18	W	CK_GATE_SAI2_CLR	0x0	Write 1 to gate clock of I2S/SAI2
19	W	CK_GATE_CPI_CLR	0x0	Write 1 to gate clock of camera parallel interface
20	W	CK_GATE_CSI2_CLR	0x0	Write 1 to gate clock of CSI-2 interface
21	W	CK_GATE_MRAM_CLR	0x0	Write 1 to gate clock of MRAM interface
22	W	CK_GATE_FILTER_CLR	0x0	Write 1 to gate clock of filter unit
23	W	CK_GATE_TIMESTAMP_CLR	0x0	Write 1 to gate clock of timestamp unit
24	W	CK_GATE_AES0_CLR	0x0	Write 1 to gate clock of AES0
25	W	CK_GATE_AES1_CLR	0x0	Write 1 to gate clock of AES1
26	W	CK_GATE_SFU_CLR	0x0	Write 1 to gate clock of SFU interface
27	W	CK_GATE_FFC0_CLR	0x0	Write 1 to gate clock of fixed/floating point converter 0
28	W	CK_GATE_FFC1_CLR	0x0	Write 1 to gate clock of fixed/floating point converter 1
29	W	CK_GATE_FFC2_CLR	0x0	Write 1 to gate clock of fixed/floating point converter 2
30	W	CK_GATE_FFC3_CLR	0x0	Write 1 to gate clock of fixed/floating point converter 3

CFG_RSTN

uDMA peripherals reset configuration (active low). At chip reset all periphs are under reset.

Bit #	R/W	Name	Reset	Description
0	R/W	RSTN_SPI0	0x0	Control reset of SPI0: 0: reset the peripheral, 1: no reset
1	R/W	RSTN_SPI1	0x0	Control reset of SPI1: 0: reset the peripheral, 1: no reset
2	R/W	RSTN_SPI2	0x0	Control reset of SPI2: 0: reset the peripheral, 1: no reset

Bit #	R/W	Name	Reset	Description
3	R/W	RSTN_SPI3	0x0	Control reset of SPI3: 0: reset the peripheral, 1: no reset
4	R/W	RSTN_UART0	0x0	Control reset of UART0: 0: reset the peripheral, 1: no reset
5	R/W	RSTN_UART1	0x0	Control reset of UART1: 0: reset the peripheral, 1: no reset
6	R/W	RSTN_UART2	0x0	Control reset of UART2: 0: reset the peripheral, 1: no reset
7	R/W	RSTN_UART3	0x0	Control reset of UART3: 0: reset the peripheral, 1: no reset
8	R/W	RSTN_UART4	0x0	Control reset of UART4: 0: reset the peripheral, 1: no reset
9	R/W	RSTN_I2C0	0x0	Control reset of I2C0: 0: reset the peripheral, 1: no reset
10	R/W	RSTN_I2C1	0x0	Control reset of I2C1: 0: reset the peripheral, 1: no reset
11	R/W	RSTN_I2C2	0x0	Control reset of I2C2: 0: reset the peripheral, 1: no reset
12	R/W	RSTN_I2C3	0x0	Control reset of I2C3: 0: reset the peripheral, 1: no reset
13	R/W	RSTN_HYPER0	0x0	Control reset of memory interface 0: 0: reset the peripheral, 1: no reset
14	R/W	RSTN_HYPER1	0x0	Control reset of memory interface 1: 0: reset the peripheral, 1: no reset
15	R/W	RSTN_JTAG	0x0	Control reset of JTAG data transfers: 0: reset the peripheral, 1: no reset
16	R/W	RSTN_SAI0	0x0	Control reset of I2S/SAI0: 0: reset the peripheral, 1: no reset
17	R/W	RSTN_SAI1	0x0	Control reset of I2S/SAI1: 0: reset the peripheral, 1: no reset
18	R/W	RSTN_SAI2	0x0	Control reset of I2S/SAI2: 0: reset the peripheral, 1: no reset
19	R/W	RSTN_CPI	0x0	Control reset of camera parallel interface: 0: reset the peripheral, 1: no reset
20	R/W	RSTN_CSI2	0x0	Control reset of CSI-2 interface: 0: reset the peripheral, 1: no reset
21	R/W	RSTN_MRAM	0x0	Control reset of MRAM interface: 0: reset the peripheral, 1: no reset
22	R/W	RSTN_FILTER	0x0	Control reset of filter unit: 0: reset the peripheral, 1: no reset
23	R/W	RSTN_TIMESTAMP	0x0	Control reset of timestamp unit: 0: reset the peripheral, 1: no reset
24	R/W	RSTN_AES0	0x0	Control reset of AES0: 0: reset the peripheral, 1: no reset
25	R/W	RSTN_AES1	0x0	Control reset of AES1: 0: reset the peripheral, 1: no reset
26	R/W	RSTN_SFU	0x0	Control reset of SFU interface: 0: reset the peripheral, 1: no reset
27	R/W	RSTN_FFC0	0x0	Control reset of fixed/floating point converter 0: 0: reset the peripheral, 1: no reset
28	R/W	RSTN_FFC1	0x0	Control reset of fixed/floating point converter 1: 0: reset the peripheral, 1: no reset
29	R/W	RSTN_FFC2	0x0	Control reset of fixed/floating point converter 2: 0: reset the peripheral, 1: no reset

Bit #	R/W	Name	Reset	Description
30	R/W	RSTN_FFC3	0x0	Control reset of fixed/floating point converter 3: 0: reset the peripheral, 1: no reset

CFG_RSTN_SET

Each bit set to 1 sets the corresponding bit in the CFG_RSTN register

Bit #	R/W	Name	Reset	Description
0	W	RSTN_SPI0_SET	0x0	Write 1 to disable reset of SPI0
1	W	RSTN_SPI1_SET	0x0	Write 1 to disable reset of SPI1
2	W	RSTN_SPI2_SET	0x0	Write 1 to disable reset of SPI2
3	W	RSTN_SPI3_SET	0x0	Write 1 to disable reset of SPI3
4	W	RSTN_UART0_SET	0x0	Write 1 to disable reset of UART0
5	W	RSTN_UART1_SET	0x0	Write 1 to disable reset of UART1
6	W	RSTN_UART2_SET	0x0	Write 1 to disable reset of UART2
7	W	RSTN_UART3_SET	0x0	Write 1 to disable reset of UART3
8	W	RSTN_UART4_SET	0x0	Write 1 to disable reset of UART4
9	W	RSTN_I2C0_SET	0x0	Write 1 to disable reset of I2C0
10	W	RSTN_I2C1_SET	0x0	Write 1 to disable reset of I2C1
11	W	RSTN_I2C2_SET	0x0	Write 1 to disable reset of I2C2
12	W	RSTN_I2C3_SET	0x0	Write 1 to disable reset of I2C3
13	W	RSTN_HYPER0_SET	0x0	Write 1 to disable reset of memory interface 0
14	W	RSTN_HYPER1_SET	0x0	Write 1 to disable reset of memory interface 1
15	W	RSTN_JTAG_SET	0x0	Write 1 to disable reset of JTAG data transfers
16	W	RSTN_SAI0_SET	0x0	Write 1 to disable reset of I2S/SAI0
17	W	RSTN_SAI1_SET	0x0	Write 1 to disable reset of I2S/SAI1
18	W	RSTN_SAI2_SET	0x0	Write 1 to disable reset of I2S/SAI2
19	W	RSTN_CPI_SET	0x0	Write 1 to disable reset of camera parallel interface
20	W	RSTN_CSI2_SET	0x0	Write 1 to disable reset of CSI-2 interface
21	W	RSTN_MRAM_SET	0x0	Write 1 to disable reset of MRAM interface
22	W	RSTN_FILTER_SET	0x0	Write 1 to disable reset of filter unit
23	W	RSTN_TIMESTAMP_SET	0x0	Write 1 to disable reset of timestamp unit
24	W	RSTN_AES0_SET	0x0	Write 1 to disable reset of AES0
25	W	RSTN_AES1_SET	0x0	Write 1 to disable reset of AES1
26	W	RSTN_SFU_SET	0x0	Write 1 to disable reset of SFU interface
27	W	RSTN_FFC0_SET	0x0	Write 1 to disable reset of fixed/floating point converter 0
28	W	RSTN_FFC1_SET	0x0	Write 1 to disable reset of fixed/floating point converter 1
29	W	RSTN_FFC2_SET	0x0	Write 1 to disable reset of fixed/floating point converter 2
30	W	RSTN_FFC3_SET	0x0	Write 1 to disable reset of fixed/floating point converter 3

CFG_RSTN_CLR

Each bit set to 1 sets the corresponding bit in the CFG_RSTN register

Bit #	R/W	Name	Reset	Description
0	W	RSTN_SPI0_CLR	0x0	Write 1 to reset SPI0
1	W	RSTN_SPI1_CLR	0x0	Write 1 to reset SPI1
2	W	RSTN_SPI2_CLR	0x0	Write 1 to reset SPI2
3	W	RSTN_SPI3_CLR	0x0	Write 1 to reset SPI3
4	W	RSTN_UART0_CLR	0x0	Write 1 to reset UART0
5	W	RSTN_UART1_CLR	0x0	Write 1 to reset UART1
6	W	RSTN_UART2_CLR	0x0	Write 1 to reset UART2
7	W	RSTN_UART3_CLR	0x0	Write 1 to reset UART3
8	W	RSTN_UART4_CLR	0x0	Write 1 to reset UART4
9	W	RSTN_I2C0_CLR	0x0	Write 1 to reset I2C0
10	W	RSTN_I2C1_CLR	0x0	Write 1 to reset I2C1
11	W	RSTN_I2C2_CLR	0x0	Write 1 to reset I2C2
12	W	RSTN_I2C3_CLR	0x0	Write 1 to reset I2C3
13	W	RSTN_HYPER0_CLR	0x0	Write 1 to reset memory interface 0
14	W	RSTN_HYPER1_CLR	0x0	Write 1 to reset memory interface 1
15	W	RSTN_JTAG_CLR	0x0	Write 1 to reset JTAG data transfers
16	W	RSTN_SAI0_CLR	0x0	Write 1 to reset I2S/SAI0
17	W	RSTN_SAI1_CLR	0x0	Write 1 to reset I2S/SAI1
18	W	RSTN_SAI2_CLR	0x0	Write 1 to reset I2S/SAI2
19	W	RSTN_CPI_CLR	0x0	Write 1 to reset camera parallel interface
20	W	RSTN_CSI2_CLR	0x0	Write 1 to reset CSI-2 interface
21	W	RSTN_MRAM_CLR	0x0	Write 1 to reset MRAM interface
22	W	RSTN_FILTER_CLR	0x0	Write 1 to reset filter unit
23	W	RSTN_TIMESTAMP_-CLR	0x0	Write 1 to reset timestamp unit
24	W	RSTN_AES0_CLR	0x0	Write 1 to reset AES0
25	W	RSTN_AES1_CLR	0x0	Write 1 to reset AES1
26	W	RSTN_SFU_CLR	0x0	Write 1 to reset SFU interface
27	W	RSTN_FFC0_CLR	0x0	Write 1 to reset fixed/floating point converter 0
28	W	RSTN_FFC1_CLR	0x0	Write 1 to reset fixed/floating point converter 1
29	W	RSTN_FFC2_CLR	0x0	Write 1 to reset fixed/floating point converter 2
30	W	RSTN_FFC3_CLR	0x0	Write 1 to reset fixed/floating point converter 3

CFG_EVENT

uDMA peripherals external event configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	CMP_EVT0	0x0	Compare event
15:8	R/W	CMP_EVT1	0x0	Compare event
23:16	R/W	CMP_EVT2	0x0	Compare event
31:24	R/W	CMP_EVT3	0x0	Compare event

FIFO_CFG

Set FIFO ID for push and pop

Bit #	R/W	Name	Reset	Description
7:0	R/W	PUSH_ID	0xFF	Sets the FIFO ID used for pushing
15:8	R/W	POP_ID	0xFF	Sets the FIFO ID used for popping

FIFO_PUSHPOP_8

Push (write) and pop (read) 8-bit word to/from the FIFO

Bit #	R/W	Name	Reset	Description
7:0	R/W	DATA	0x00	A write pushes an 8-bit data to the FIFO, a read pops an 8-bit data from the FIFO

FIFO_PUSHPOP_16

Push (write) and pop (read) 16-bit word to/from the FIFO

Bit #	R/W	Name	Reset	Description
15:0	R/W	DATA	0x00	A write pushes a 16-bit data to the FIFO, a read pops a 16-bit data from the FIFO

FIFO_PUSHPOP_24

Push (write) and pop (read) 24-bit word to/from the FIFO

Bit #	R/W	Name	Reset	Description
23:0	R/W	DATA	0x00	A write pushes a 24-bit data to the FIFO, a read pops a 24-bit data from the FIFO

FIFO_PUSHPOP_32

Push (write) and pop (read) 32-bit word to/from the FIFO

Bit #	R/W	Name	Reset	Description
31:0	R/W	DATA	0x00	A write pushes a 32-bit data to the FIFO, a read pops a 32-bit data from the FIFO

DATAMOVE_CFG

Configure data movement

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID_0	0xFF	Sets the source ID used by the data mover
15:8	R/W	DEST_ID_0	0xFF	Sets the destination ID used by the data mover
23:16	R/W	SOURCE_ID_1	0xFF	Sets the source ID used by the data mover
31:24	R/W	DEST_ID_1	0xFF	Sets the destination ID used by the data mover

DATAMOVE0_SIZE

Control of mover channel 0

Bit #	R/W	Name	Reset	Description
20:0	R/W	SIZE	0x0	Write sets the number of bytes to be moved. Read returns the number of bytes remaining
30	R/W	STOP	0x0	When written to 1 stops the data mover. When read return the enable status of the data mover
31	R/W	EN	0x0	When written to 1 enables the data mover. When read return the enable status of the data mover

DATAMOVE1_SIZE

Control of mover channel 1

Bit #	R/W	Name	Reset	Description
20:0	R/W	SIZE	0x0	Write sets the number of bytes to be moved. Read returns the number of bytes remaining
30	R/W	STOP	0x0	When written to 1 stops the data mover. When read return the enable status of the data mover
31	R/W	EN	0x0	When written to 1 enables the data mover. When read return the enable status of the data mover

STREAM_CFG

Configure blocking behavior of streams

Bit #	R/W	Name	Reset	Description
0	R/W	BLK_STREAM0	0x0	Blocking state for stream 0: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM1	0x0	Blocking state for stream 1: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM2	0x0	Blocking state for stream 2: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM3	0x0	Blocking state for stream 3: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM4	0x0	Blocking state for stream 4: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM5	0x0	Blocking state for stream 5: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM6	0x0	Blocking state for stream 6: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM7	0x0	Blocking state for stream 7: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM8	0x0	Blocking state for stream 8: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM9	0x0	Blocking state for stream 9: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM10	0x0	Blocking state for stream 10: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM11	0x0	Blocking state for stream 11: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM12	0x0	Blocking state for stream 12: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM13	0x0	Blocking state for stream 13: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM14	0x0	Blocking state for stream 14: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available
0	R/W	BLK_STREAM15	0x0	Blocking state for stream 15: 0: non blocking, 1: blocking, i.e. “ready” asserted only when data available

TIMEOUT_PRE0

Configuration of the frequency prescaler for timeout ch0

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH0

Configuration for timeout ch0

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE1

Configuration of the frequency prescaler for timeout ch1

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH1

Configuration for timeout ch1

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE2

Configuration of the frequency prescaler for timeout ch2

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH2

Configuration for timeout ch2

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE3

Configuration of the frequency prescaler for timeout ch3

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH3

Configuration for timeout ch3

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE4

Configuration of the frequency prescaler for timeout ch4

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH4

Configuration for timeout ch4

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE5

Configuration of the frequency prescaler for timeout ch5

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH5

Configuration for timeout ch5

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE6

Configuration of the frequency prescaler for timeout ch6

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH6

Configuration for timeout ch6

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

TIMEOUT_PRE7

Configuration of the frequency prescaler for timeout ch7

Bit #	R/W	Name	Reset	Description
15:0	R/W	CNT	0x0	Set the target for the timeout counter
16	R/W	EN	0x0	Enable/disable the timeout prescaler: 0: disabled, 1: enabled
17	W	CLR	0x0	Reset the timeout prescaler to 0

TIMEOUT_CH7

Configuration for timeout ch7

Bit #	R/W	Name	Reset	Description
7:0	R/W	SOURCE_ID	0x0	Set the uDMA source ID used by the timeout
9:8	R/W	MODE	0x0	Set the mode to start/stop the timeout: b00: software triggered, b01: started by start of transfer and stopped by end of transfer, b10: counter cleared at each data RX/TX
10	R/W	EN	0x0	Enable/disable the timeout: 0: disabled, 1: enabled
31:16	R/W	CNT	0x0	Set the target for the timeout counter

5.3.2 SPI Interface

SPI master and slave interface, which features:

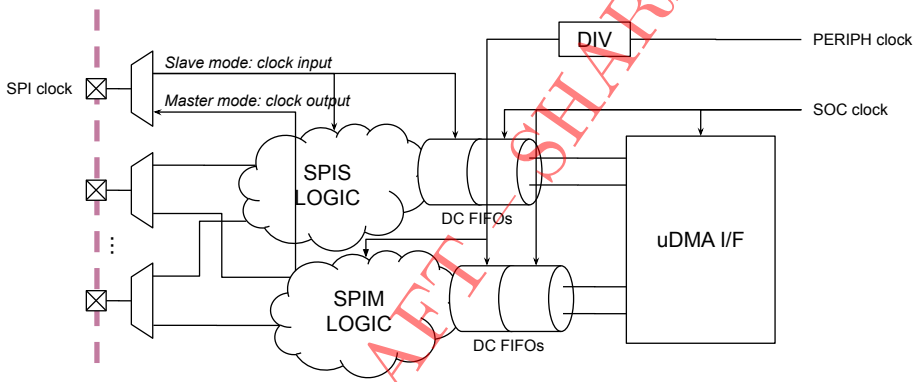
- Master SPI/QuadSPI with flexible configuration.
- Slave SPI – only supports 1-bit MISO(SDO0)/MOSI(SDI1), MSB first, CPHA=0, CPOL=0, 8-bit transfer.
- TX/RX/full duplex transfers.
- Stream of 32-bit commands to program complex transfers.
- Optional chip select, active low.

5.3.2.1 Clocking

The figure below shows the clocking scheme of the SPI peripheral. This peripheral embeds both an SPI/quadSPI master and an SPI slave block. One or the other is used depending on the used mode.

The master mode logic is clocked by the PERIPH clock generated by the FLL, on which an integer division may be applied. This clock is also propagated to the external component. The slave mode logic is clocked by the external clock driven by the distant peripheral.

Note that the SPI configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.2.2 SPI Behavior

The SPI slave uses three UDMA streams:

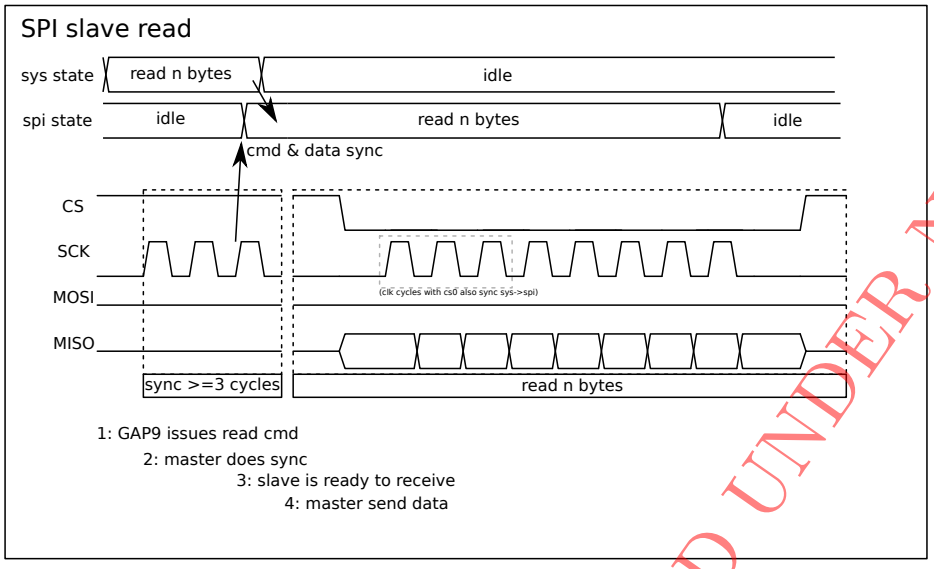
- Two TX streams to send command and data, respectively.
- One RX stream to receive data.

To read or write data on the SPI bus, a series of commands must be sent, also referred to as SPI micro-code, detailed later in this section. For instance, the command SPI_CMD_RX_DATA prepares the SPI slave to read some bytes on the bus, and the command SPI_CMD_TX_DATA prepares it to write similarly.

5.3.2.3 SPI Slave Clocking

As the SPI slave uses its own clock domain derived from the SPI bus clock, incoming commands from GAP9 (clocked on SOC clock) need to cross dual clock FIFOs to actually enter the SPI slave block. This requires that the SPI bus clock toggles for at least 3 cycles. When initiating a transfer, this means that the clock should toggle for at least 3 cycles before asserting the chip select low, so that the command for this transfer is properly taken into account.

However, as soon as the SPI clock is toggling, commands can enter the SPI slave block when chip select is high or low, including when a read/write command is being processed – this allows to prepare for the next read/write command. If the SPI master attempts to read/write a byte but the SPI slave has no commands to process, this "command-less" byte is ignored by the SPI slave. This property allows to properly start a transfer in cases when the master does not have precise control of the SPI clock ahead of the chip select assersion: it first sends a "dummy" byte, which allows the command to enter SPI slave, then the useful payload can be sent and/or received.



5.3.2.4 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_DEST	0x0	32	Stream ID for the RX uDMA channel
TX_DEST	0x4	32	Stream ID for the TX uDMA channel
CMD_DEST	0x8	32	Stream ID for the CMD uDMA channel
STATUS	0x30	32	Status register
CONFIG	0x34	32	Configuration register

RX_DEST

Stream ID for the RX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_DEST	0xFF	Stream ID for the RX uDMA channel. Default is 0xFF (channel disabled).

TX_DEST

Stream ID for the TX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	TX_DEST	0xFF	Stream ID for the TX uDMA channel. Default is 0xFF (channel disabled).

CMD_DEST

Stream ID for the CMD uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	CMD_DEST	0xFF	Stream ID for the CMD uDMA channel. Default is 0xFF (channel disabled).

STATUS

Status register

Bit #	R/W	Name	Reset	Description
3:0	R/W	STATUS	0	Misc status
4	R/W	CMD_EVENT	0	Is set to 1 upon transmission of a command; write register to clear
5	R/W	RX_OVERFLOW_-EVENT	0	(slave only) Set to 1 in case of RX overflow; write register to clear
6	R/W	TX_UNDERFLOW_-EVENT	0	(slave only) Set to 1 in case of TX underflow; write register to clear

CONFIG

Configuration register

Bit #	R/W	Name	Reset	Description
0	R/W	SPI_SLAVE_MODE	1	Enable SPI slave mode
1	R/W	RX_OVRFLW_EN	1	Enable RX overflow event
2	R/W	TX_UNDRFLW_EN	1	Enable TX underflow event

5.3.2.5 SPI micro-code

Command name	Width	Command code	Description
SPI_CMD_CFG	32	0x0	(Master only) Set the configuration for the SPI Master IP
SPI_CMD_SOT	32	0x1	(Master only) Select the Chip Select (CS)
SPI_CMD_SEND_CMD	32	0x2	(Master only) Transmit up to 16 bits of data in the command word
SPI_CMD_DUMMY	32	0x4	(Master only) Receive a number of dummy bits (not sent to the RX interface)
SPI_CMD_WAIT	32	0x5	(Master only) Wait for an external event to move to the next instruction
SPI_CMD_TX_DATA	32	0x6	Transfer from master to slave (Master: send data; slave: receive data) – max 64 kwords
SPI_CMD_RX_DATA	32	0x7	Transfer from slave to master (Master: receive data; slave: send data) – max 64 kwords
SPI_CMD_RPT	32	0x8	(Master only) Repeat the commands from here to SPI_CMD_RPT_END command a given number of times
SPI_CMD_EOT	32	0x9	Clear the Chip Select (CS) after transfer
SPI_CMD_RPT_END	32	0xA	(Master only) End of the repeated command section
SPI_CMD_RX_CHECK	32	0xB	(Master only) Check up to 16 bits of data against an expected value
SPI_CMD_FULL_DUPLEX	32	0xC	Transfer in full duplex mode (send and receive data) – max 64 kwords
SPI_CMD_SETUP_AG	32	0xD	Setup a register of the address generator (the register value is given in the following word)

SPI_CMD_CFG

Bit #	Name	Description
7:0	CLKDIV	Clock divider value
8	CPHA	Clock phase (CPHA) value
9	CPOL	Clock polarity (CPOL) value
31:28	SPI_CMD	Command code – here “CFG”=0x0

SPI_CMD_SOT

Bit #	Name	Description
1:0	CS	Select the Chip Select (CS): b00: csn0; b01: csn1; b10: csn2; b11: csn3
31:28	SPI_CMD	Command code – here “SOT”=0x1

SPI_CMD_SEND_CMD

Bit #	Name	Description
15:0	DATA_VALUE	Value to be sent through the command. MSB must always be at bit 15, even if less than 16 bits are to be sent
19:16	DATA_SIZE	Size in bits of the value to be sent, minus 1
27	QPI	Set to 1 to use quadSPI
31:28	SPI_CMD	Command code – here “SEND_CMD”=0x2

SPI_CMD_DUMMY

Bit #	Name	Description
20:16	DUMMY_CYCLE	Number of dummy cycles to perform
27	QPI	Set to 1 to use quadSPI
31:28	SPI_CMD	Command code – here “DUMMY”=0x4

SPI_CMD_WAIT

Bit #	Name	Description
6:0	EVENT_ID_CYCLE_-COUNT	External event id or Number of wait cycles
9:8	WAIT_TYPE	Type of wait: b00: wait for a soc event selected by EVENT_-ID; ’b01: wait for CYCLE_COUNT cycles; b10: reserved; b11: reserved
31:28	SPI_CMD	Command code – here “WAIT”=0x5

SPI_CMD_TX_DATA

Bit #	Name	Description
15:0	WORD_NUM	Number of words (max 64K) to send (master) or receive (slave), minus 1. The number of bits sent depends on the word size.
16	IGNORE_SELECT	(Slave only) Set to 1 to ignore slave select
20:16	WORD_SIZE	(Master only) Size in bits, minus 1, of data words to send. Each word is read from L2 with a transfer of a configurable size (see WORD_PER_TRANSF field).
22:21	WORD_PER_TRANSF	(Master only) Number of bytes transferred from L2 at each transfer: b00: 1 byte per transfer; 01: 2 bytes per transfer; b10: 4 bytes per transfer; b11: reserved. Using unaligned values between WORD_SIZE and WORD_PER_TRANSF can be used to insert dummy bits and drop bits, compared to L2 content.
26	LSB	(Master only) Set to 1 to use least-significant-bit-first transfers
27	QPI	(Master only) Set to 1 to use quadSPI
31:28	SPI_CMD	Command code – here “TX_DATA”=0x6

SPI_CMD_RX_DATA

Bit #	Name	Description
15:0	WORD_NUM	Number of words (max 64K) to receive master) or send (slave), minus 1. The number of bits received depends on the word size.
16	IGNORE_SELECT	(Slave only) Set to 1 to ignore slave select
20:16	WORD_SIZE	(Master only) Size in bits, minus 1, of data words to receive.
22:21	WORD_PER_TRANSF	(Master only) Number of received words transferred to L2 in a single transfer: b00: 1 word per transfer; 01: 2 words per transfer; b10: 4 words per transfer; b11: reserved. A uDMA transfer to L2 is always 32-bit. To build it, WORD_SIZE is rounded-up to the closest supported word size (8, 16 or 32 bits), and the configured number of words are then packed. If the rounded word size multiplied by this field is not equal to 32 bits, then either dummy bits are inserted, or exceeding bits are dropped.
26	LSB	(Master only) Set to 1 to use least-significant-bit-first transfers
27	QPI	(Master only) Use quadSPI mode
31:28	SPI_CMD	Command code – here “RX_DATA”=0x7

SPI_CMD_RPT

Bit #	Name	Description
15:0	RPT_CNT	Number of repeat iterations (max 64K)
31:28	SPI_CMD	Command code – here “RPT”=0x8

SPI_CMD_EOT

Bit #	Name	Description
0	EVENT_GEN	Enable generation of EOT events: 0: disabled; 1: enabled
1	CS_KEEP	Keep chip-select active: 0: CS is released (high) at the end of the transfer: 1: CS is kept active (low) after the transfer is finished
31:28	SPI_CMD	Command code – here “EOT”=0x9

SPI_CMD_RPT_END

Bit #	Name	Description
31:28	SPI_CMD	Command code – here “RPT_END”=0xA

SPI_CMD_RX_CHECK

Bit #	Name	Description
15:0	COMP_DATA	Data value to compare (max 16 bits)
19:16	STATUS_SIZE	Size in bits of the value to check, minus 1
25:24	CHECK_TYPE	Type of check: b00: compare all bits; b01: compare only ones; b10: compare only zeros; b11: reserved
26	LSB	Set to 1 to use least-significant-bit-first transfers
27	QPI	Set to 1 to use quadSPI
31:28	SPI_CMD	Command code – here “RX_CHECK”=0xB

SPI_CMD_FULL_DUPLEX

Bit #	Name	Description
15:0	WORD_NUM	Number of words (max 64K) to send and receive, minus 1. The number of bits received and sent depends on the word size.
16	IGNORE_SELECT	(Slave only) Set to 1 to ignore slave select
20:16	WORD_SIZE	(Master only) Size in bits, minus 1, of data words to send and receive
22:21	WORD_PER_TRANSF	(Master only) Number of words transferred from L2 at each transfer: b00: 1 byte per transfer; 01: 2 bytes per transfer; b10: 4 bytes per transfer; b11: reserved. A uDMA transfer to L2 is always 32-bit. To build it, WORD_SIZE is rounded-up to the closest supported word size (8, 16 or 32 bits), and the configured number of words are then packed. If the rounded word size multiplied by this field is not equal to 32 bits, then either dummy bits are inserted, or exceeding bits are dropped.
26	LSB	(Master only) Set to 1 to use least-significant-bit-first transfers
31:28	SPI_CMD	Command code – here “FULL_DUPLEX”=0xC

SPI_CMD_SETUP_AG

Bit #	Name	Description
1:0	REG_SELECT	Register address to set. Configuration data is taken in the following word.
8	STREAM_SELECT	Selects which address generator to set: 0: use stream ID of the RX channel; 1: use stream ID of the TX channel
31:28	SPI_CMD	Command code – here “SETUP_AG”=0xD

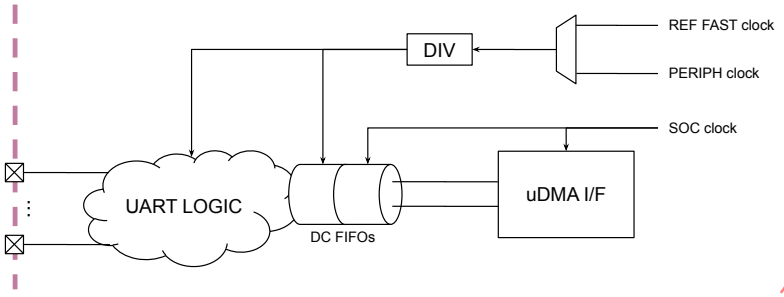
5.3.3 UART Interface

GAP9 UART peripherals offer the following features:

- The usual standard full duplex UART interface.
- Configurable baudrate.
- 5, 6, 7 or 8 data bits.
- 0 or 1 parity bit.
- 1 or 2 stop bits.
- Optional hardware control flow using rtsn_o and ctsn_i signals, with configurable FIFO high limit (allow receiving words after rtsn_o is deasserted).
- Master synchronous mode.
- Configurable clock phase and polarity.

5.3.3.1 Clocking

The figure below shows the clocking scheme of the UART peripheral. The clock may be chosen between the PERIPH clock generated by the FLL and the REF clock. An integer division may then be applied on the selected clock. Note that the UART configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.3.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_DEST	0x0	32	Stream ID for the uDMA RX channel
TX_DEST	0x4	32	Stream ID for the uDMA TX channel
MISC	0x10	32	Send events
STATUS	0x20	32	Status register
SETUP	0x24	32	Configuration register
ERROR	0x28	32	Error register, cleared on reading
IRQ_EN	0x2C	32	Enable/disable events
SETUP_2	0x38	32	Configuration register 2
REF_CLK_MUX	0x3C	32	Clock selection for the UART

RX_DEST

Stream ID for the uDMA RX channel

Bit #	R/W	Name	Reset	Description
7:0	RW	RX_DEST	0xFF	Stream ID for the RX uDMA channel. Default is 0xFF (channel disabled)

TX_DEST

Stream ID for the uDMA TX channel

Bit #	R/W	Name	Reset	Description
7:0	RW	TX_DEST	0xFF	Stream ID for the TX uDMA channel. Default is 0xFF (channel disabled)

MISC

Send events

Bit #	R/W	Name	Reset	Description
0	W	TX_FIFO_CLEAR_-EVENT_O	0	Send event to clear TX FIFO
1	W	RX_FIFO_CLEAR_-EVENT_O	0	Send event to clear RX FIFO
2	W	TX_FSM_RESET_-EVENT_O	0	Send event to reset TX FSM
3	W	RX_FSM_RESET_-EVENT_O	0	Send event to reset RX FSM

STATUS

Status register

Bit #	R/W	Name	Reset	Description
0	R	TX_BUSY	0	Transmitter is sending a frame
1	R	RX_BUSY	0	Receiver is receiving a frame

SETUP

Configuration register

Bit #	R/W	Name	Reset	Description
0	RW	PARITY_ENA	0	Enable parity bit for TX and RX blocks
2:1	RW	BIT_LENGTH	0	RX/TX word width (see encoding below)
3	RW	STOP_BITS	0	Stop bits count (see encoding below)
8	RW	TX_ENA	0	Enable transmitter
9	RW	RX_ENA	0	Enable receiver
10	RW	CTS_EN	0	Flow control: enable Clear To Send input pin. Transmitter will send next word if UART CTS input is 0.
11	RW	RTS_EN	0	Flow control: enable Ready To Send output pin. UART RTS output is set to 0 if the receiver can receive next word.
12	RW	TX_CLK_EN	0	Enable synchronous master mode
13	RW	TX_CLK_POL	0	Configure TX clock polarity (see encoding below)
14	RW	TX_CLK_PHA	0	Configure TX clock phase (see encoding below)
31:16	RW	CLKDIV	0	Baudrate divider applied to selected internal clock. Baudrate = Clk_freq / (CLKDIV + 1)

ERROR

Error register, cleared on reading

Bit #	R/W	Name	Reset	Description
0	R	ERR_OVERFLOW	0	RX overflow flag
1	R	ERR_PARITY	0	RX parity error flag

IRQ_EN

Enable/disable events

Bit #	R/W	Name	Reset	Description
0	R/W	RX_IRQ	0	Emit event if RX received a word
1	R/W	ERR_IRQ	0	Emit event on an error (see ERROR register)
2	R/W	TX_IRQ	0	Emit event after a byte is sent, after stop symbol is transmitted

SETUP_2

Configuration register 2

Bit #	R/W	Name	Reset	Description
3:0	R/W	RTS_HIGH_LIMIT	4	Deassert UART RTS when number of data in the FIFO $\geq$ RTS_HIGH_LIMIT. FIFO size is 8.

REF_CLK_MUX

Clock selection for the UART

Bit #	R/W	Name	Reset	Description
0	R/W	SEL_REF_FAST_CLK	0	Set to 1 to use REF FAST clock as IP clock instead of PERIPH clock

5.3.3.3 Encoding of SETUP Register Fields

BIT_LENGTH field	TX/RX word width (bits)
0	5
1	6
2	7
3	8

STOP_BITS field	Number of stop bits
0	1
1	2

TX_CLK_POL field	TX_CLK_PHA field	TX clock value when idle	TX clock sampling edge
0	0	0	rising
0	1	0	falling
1	0	1	falling
1	1	1	rising

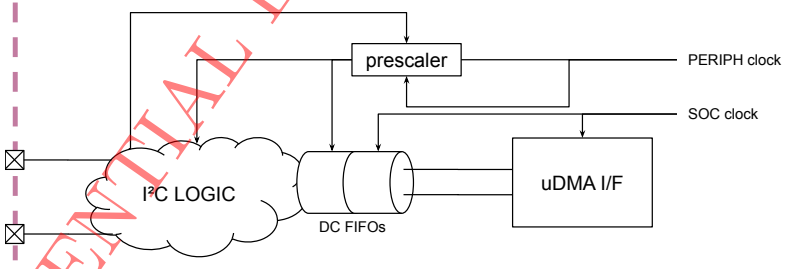
5.3.4 I2C Interface

GAP9 I2C peripherals offer the following features:

- can be simultaneously an I2C leader (master) and follower (slave)
- supports multimaster mode
- configurable clock period & duty cycle
- clock stretching
- supports standard mode (100kHz), Fast Mode (400kHz) and FM+ (1MHz)
- frame error checking
- arbitration loss error checking
- pulse filtering & 3-bit error correction
- leader:
  - NACK error check
  - NACK error optionally generates a stop
- follower:
  - 2-address slots
    - configurable event on receive/send on start-of-frame/end-of-frame
    - 7-/10-bit mode each
    - 5-bit address mask

5.3.4.1 Clocking

The figure below shows the clocking scheme of the I2C peripheral. The peripheral is clocked by the PERIPH clock generated by the FLL, on which an integer division may be applied. Note that the I2C configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.4.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
FOLL_UDMA_RX_DEST_REG_IDX	0x0	32	I2C follower RX uDMA channel
FOLL_UDMA_TX_DEST_REG_IDX	0x4	32	I2C follower TX uDMA channel
UDMA_CMD_DEST_REG_IDX	0x8	32	Command TX uDMA channel
LEAD_UDMA_RX_DEST_REG_IDX	0xC	32	I2C leader RX uDMA channel
LEAD_UDMA_TX_DEST_REG_IDX	0x10	32	I2C leader TX uDMA channel
STATUS_REG_IDX	0x14	32	Incoming event flag / emit event

FOLL_UDMA_RX_DEST_REG_IDX

I2C follower RX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	UDMA_STREAM_ID	0xFF	UDMA channel ID (0xFF=disabled)

FOLL_UDMA_TX_DEST_REG_IDX

I2C follower TX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	UDMA_STREAM_ID	0xFF	UDMA channel ID (0xFF=disabled)

UDMA_CMD_DEST_REG_IDX

Command TX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	UDMA_STREAM_ID	0xFF	UDMA channel ID (0xFF=disabled)

LEAD_UDMA_RX_DEST_REG_IDX

I2C leader RX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	UDMA_STREAM_ID	0xFF	UDMA channel ID (0xFF=disabled)

LEAD_UDMA_TX_DEST_REG_IDX

I2C leader TX uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	UDMA_STREAM_ID	0xFF	UDMA channel ID (0xFF=disabled)

STATUS_REG_IDX

Incoming event flag / emit event

Bit #	R/W	Name	Reset	Description
0	R/W	STATUS_FOLL_SOF_-RCV_EVENT_I_IDX	0	Follower is addressed by a read. Write 1 to clear.
1	R/W	STATUS_FOLL_SOF_-SND_EVENT_I_IDX	0	Follower is addressed by a write. Write 1 to clear.
2	R/W	STATUS_FOLL_-EOF_RCV_EVENT_-I_IDX	0	Follower is addressed by a read which closes transfer. Write 1 to clear.
3	R/W	STATUS_FOLL_-EOF_SND_EVENT_-I_IDX	0	Follower is addressed by a write which closes transfer. Write 1 to clear.
4	R/W	STATUS_FOLL_-ERROR_ARLO_-EVENT_I_IDX	0	Follower is addressed and received an arbitration loss. Write 1 to clear.
5	R/W	STATUS_FOLL_ER-ROR_FRAMING_-EVENT_I_IDX	0	Follower is addressed and noticed incorrect framing. Write 1 to clear.
14	W	STATUS_FOLL_UN-LOCK_EVENT_O_-IDX	0	Write 1 to unlock follower TX RX channels after a SOF/EOF/ERROR event
15	W	STATUS_FOLL_-PURGE_EVENT_O_-IDX	0	Write 1 to purge TX RX channels FIFOs of any remaining data
16	R/W	STATUS_LEAD_-ERROR_NACK_-EVENT_I_IDX	0	Leader encountered an unexpected NACK. Write 1 to clear.
17	R/W	STATUS_LEAD_-ERROR_ARLO_-EVENT_I_IDX	0	Leader encountered an arbitration loss. Write 1 to clear.
18	R/W	STATUS_LEAD_ER-ROR_FRAMING_-EVENT_I_IDX	0	Leader encountered an incorrect framing. Write 1 to clear.
19	R/W	STATUS_LEAD_COM-MAND_EVENT_I_IDX	0	Leader emitted a command event. Does not lock. Write 1 to clear.
22	W	STATUS_LEAD_UN-LOCK_EVENT_O_-IDX	0	Write 1 to unlock leader TX RX channels after a SOF/EOF/ERROR event
23	W	STATUS_LEAD_-PURGE_EVENT_O_-IDX	0	Write 1 to purge cmd TX RX leader FIFOs of any remaining data
24	W	STATUS_I2C_SOFT_-RESET_EVENT_O_-IDX	0	Write 1 to soft reset I2C, if it is deadlocked due to bus error
25	W	STATUS_I2C_-PRESCALER_SET_-DIV10_EVENT_O_-IDX	0	Write 1 to set prescaler to divide by 10 instead of 100 by default

5.3.4.3 I2C micro-code

Command name	Width	Command code	Description
CMD_MISC_NOP	32	0x00	Does nothing
CMD_MISC_WAIT	32	0x01	Wait one I2C clock cycle (repeated if REPEAT_CNT > 1)
CMD_MISC_REPEAT	32	0x02	Reload the 16-bit repeat downcounter
CMD_MISC_WAIT_I2C_-PERIOD_END	32	0x03	Wait one I2C SCL period cycle - see CMD_TIMING (repeated if REPEAT_CNT > 1)
CMD_TIMING	32	0x10	Setup I2C_CLOCK period and I2C SCL low/high delays
CMD_FOLL_ADDR	32	0x20	Setup follower adressing & events
CMD_LEAD_START	32	0x30	Send a (re)start condition
CMD_LEAD_SEND	32	0x31	Send a byte from TX stream, check ACK (repeated if REPEAT_CNT > 1)
CMD_LEAD_SEND_IMM	32	0x32	Send a byte from the command, check ACK
CMD_LEAD_SEND_-IMM_ADDR	32	0x37	Send an addr7 or addr10, check ACK
CMD_LEAD_RECV	32	0x33	Receives a byte that is not the last byte, send ACK (repeated if REPEAT_CNT > 1)
CMD_LEAD_RECV_LAST	32	0x34	Receives a byte that is the last byte, send NACK (repeated if REPEAT_CNT > 1, in this case the first REPEAT_CNT-1 bytes are ACKed)
CMD_LEAD_STOP	32	0x36	Generates a stop condition
CMD_EVENT_RECV	32	0x40	Wait for an external event of a given index (repeated if REPEAT_CNT > 1)
CMD_EVENT_SEND	32	0x41	Triggers an internal 'command event'
CMD_UDMA_TX_-CHAN_CFG	32	0x50	Send a configuration command to uDMA TX channel (the register value is given in the following word)
CMD_UDMA_RX_-CHAN_CFG	32	0x51	Send a configuration command to uDMA RX channel (the register value is given in the following word)

CMD_MISC_NOP

Bit #	Name	Description
31:24	I2C_CMD	Command code – here "MISC_NOP"=0x00

CMD_MISC_WAIT

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “MISC_WAIT”=0x01

CMD_MISC_REPEAT

Bit #	Name	Description
15:0	SET_REPEAT_CNT	Set REPEAT_CNT. Commands that can be repeated will be repeated SET_REPEAT_CNT times, after which REPEAT_CNT will be equal to 1.
31:24	I2C_CMD	Command code – here “MISC_REPEAT”=0x02

CMD_MISC_WAIT_I2C_PERIOD_END

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “MISC_WAIT_I2C_PERIOD_END”=0x03

CMD_TIMING

Bit #	Name	Description
3:0	DL	Setup SCL low duration: T_low=i2c_period × (DL + 4)
7:4	DH	Setup SCL high duration: T_high=i2c_period × (DL + 4)
11:8	DP	Division factor between PERIPH clock and I2C clock, minus 1. Must be ≥ 1. DP=99 at reset (i.e. divide by 100).
31:24	I2C_CMD	Command code – here “TIMING”=0x10

CMD_FOLL_ADDR

Bit #	Name	Description
8:0	ADDR_MATCH	Address of the follower
14:10	ADDR_MASK	Address mask (set to b11111 to match a single address)
15	ENABLE_7BIT_ADDR	Set to 1 to enable 7-bit address matching
16	ENABLE_10BIT_ADDR	Set to 1 to enable 10-bit address matching
17	ENABLE_SOF_RCV	Set to 1 to enable SOF event when receiving
18	ENABLE_EOF_RCV	Set to 1 to enable SOF event when receiving
19	ENABLE_SOF_SND	Set to 1 to enable EOF event when sending
20	ENABLE_EOF_SND	Set to 1 to enable EOF event when sending
21	ADDR_PUSH_ENABLE	Enable incoming addressing mode from leader to be pushed into RX channel
23:22	ADDR_SLOT_IDX	Address match slot to be configured
31:24	I2C_CMD	Command code – here “FOLL_ADDR”=0x20

CMD_LEAD_START

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “LEAD_START”=0x30

CMD_LEAD_SEND

Bit #	Name	Description
22	TWEAK_IGNORE_NACK	If 1, ignore NACK error: it will not trigger an event nor a lock
23	TWEAK_STOP_ON_NACK	If 1, on NACK automatically send a stop
31:24	I2C_CMD	Command code – here “LEAD_SEND”=0x31

CMD_LEAD_SEND_IMM

Bit #	Name	Description
7:0	DATA	Data to be sent
22	TWEAK_IGNORE_NACK	If 1, ignore NACK error: it will not trigger an event nor a lock
23	TWEAK_STOP_ON_NACK	If 1, on NACK automatically send a stop
31:24	I2C_CMD	Command code – here “LEAD_SEND_IMM”=0x32

CMD_LEAD_SEND_IMM_ADDR

Bit #	Name	Description
7:0	ADDR8	Address LSBs: in 7-bit addressing, it is made of the 7 address bits + direction bit; in 10-bit adressing, it is the 8 LSBs of the address.
15:8	ADDR16	Address MSBs in 10-bit addressing: b11110 + address bits 9 and 8 + direction bit.
15	ADDRESS_MODE	If 0, send 7-bit address (i.e. send ADDR8 byte only); if 1, send 10-bit address (i.e. ADDR16 as the first byte, then ADDR8 as the second byte).
22	TWEAK_IGNORE_NACK	If 1, ignore NACK error: it will not trigger an event nor a lock
23	TWEAK_STOP_ON_NACK	If 1, on NACK automatically send a stop
31:24	I2C_CMD	Command code – here “LEAD_SEND_IMM_ADDR”=0x37

CMD_LEAD_RECV

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “LEAD_RECV”=0x33

CMD_LEAD_RECV_LAST

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “LEAD_RECV_LAST”=0x34

CMD_LEAD_STOP

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “LEAD_STOP”=0x36

CMD_EVENT_RECV

Bit #	Name	Description
1:0	EVENT_IDX	Index of expected external event
31:24	I2C_CMD	Command code – here “EVENT_RECV”=0x40

CMD_EVENT_SEND

Bit #	Name	Description
31:24	I2C_CMD	Command code – here “EVENT_SEND”=0x41

CMD_UDMA_TX_CHAN_CFG

Bit #	Name	Description
1:0	TX_REG_SELECT	TX register address to set. Configuration data is taken in the following word.
31:24	I2C_CMD	Command code – here “UDMA_TX_CHAN_CFG”=0x50

CMD_UDMA_RX_CHAN_CFG

Bit #	Name	Description
1:0	RX_REG_SELECT	RX register address to set. Configuration data is taken in the following word.
31:24	I2C_CMD	Command code – here “UDMA_RX_CHAN_CFG”=0x51

5.3.5 External Memory Controller

The memory controller manages the following features:

- Controls all HYPERBUS bus specific sequencing, protocol, arbitration and timing.
  - Supports RAM and Flash memories types.
  - Support non-aligned access, the minimum access unit is 1 byte except for HYPER Flash write access (2 bytes).
  - Support maximum address size up to 31 bits (ADDR[0:30]), the highest bit ADDR[31] is used to discriminate register accesses.
- Supports Octo SPI and Single SPI.

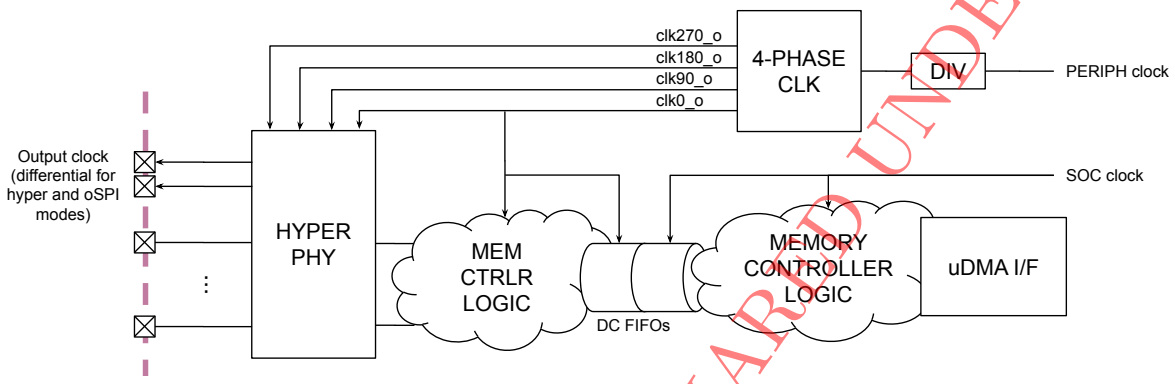
  - For Octo SPI, it supports both DTR and STR mode for all stages CMD, ADDR, ALTER, DATA. CMD size is 0-2 bytes, (ADDRESS + ALTER) size is 0-6 bytes in which ALTER is up to 2 bytes. So the total control region is up to 8 bytes with cmd_size[1:0] and addr_size[2:0] signals. The ALTER bytes are combined with ADDR bytes, and when (ADDRESS + ALTER) size is less than or equal to 4 bytes, use only EXT/OSPI_ADDR[31:0] in priority (see examples in the table below).
- Support user controlled chip select.

CMD size	ADDR size	ALTER size	OSPI_CMD[15:0]	EXT/OSPI_ADDR[31:0]	OSPI_ALTER[15:0]
0 (*)	0 (*)	0 (*)			
1	0	0	(CMD << 8)		
1	0	1	(CMD << 8)	(ALTER << 24)	
1	0	2	(CMD << 8)	(ALTER << 16)	
1	1	0	(CMD << 8)	(ADDR << 24)	
1	1	1	(CMD << 8)	(ADDR << 24) or (ALTER << 16)	
1	1	2	(CMD << 8)	(ADDR << 24) or (ALTER << 8)	
1	2	0	(CMD << 8)	(ADDR << 16)	
1	2	1	(CMD << 8)	(ADDR << 16) or (ALTER << 8)	
1	2	2	(CMD << 8)	(ADDR << 16) or (ALTER)	
1	3	0	(CMD << 8)	(ADDR << 8)	
1	3	1	(CMD << 8)	(ADDR << 8) or (ALTER)	
1	3	2	(CMD << 8)	(ADDR << 8) or (ALTER[15:7])	(ALTER[7:0]
1	4	0	(CMD << 8)	ADDR	
1	4	1	(CMD << 8)	ADDR	(ALTER << 8)
1	4	2	(CMD << 8)	ADDR	ALTER
2	0	0	CMD		
2	0	1	CMD	(ALTER << 24)	
2	0	2	CMD	(ALTER << 16)	
2	1	0	CMD	(ADDR << 24)	
2	1	1	CMD	(ADDR << 24) or (ALTER << 16)	
2	1	2	CMD	(ADDR << 24) or (ALTER << 8)	
2	2	0	CMD	(ADDR << 16)	
2	2	1	CMD	(ADDR << 16) or (ALTER << 8)	
2	2	2	CMD	(ADDR << 16) or (ALTER)	
2	3	0	CMD	(ADDR << 8)	
2	3	1	CMD	(ADDR << 8) or (ALTER)	
2	3	2	CMD	(ADDR << 8) or (ALTER[15:7])	(ALTER[7:0]
2	4	0	CMD	ADDR	
2	4	1	CMD	ADDR	(ALTER << 8)
2	4	2	CMD	ADDR	ALTER

(*) Only with data load

5.3.5.1 Clocking

The figure below shows the clocking scheme of the memory controller peripheral. The peripheral is clocked by the PERIPH clock generated by the FLL, on which an integer division may be applied. Four clocks are then derived, with 0°, 90°, 180° and 270° phase shift, and provided to the PHY. The 0-shifted clock is used for the controller logic. Depending on the used mode, the clock is sent to the external memory component, either as a differential signal – for hyper and oSPI – or as a single lane signal – for qSPI and SDIO. When receiving data in hyper and oSPI modes, the read/write data strobe (RWDS) signal is a shifted version of this clock, sent by the external memory component, and edge-aligned with received data. Note that the memory controller configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.5.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_DEST	0x0	32	Stream ID for the uDMA channel
TX_DEST	0x4	32	Stream ID for the uDMA channel
TRANS_MODE	0x8	32	Configure transaction mode
TRANS_ADDR	0xC	32	Configure each transaction addr
TRANS_SIZE	0x10	32	Configure each transaction size
TRANS_CFG	0x14	32	Start each transaction rx/tx
OSPI_CMD_XIP	0x18	32	OSPI XIP command and psram command config
OSPI_CFG_XIP	0x1C	32	OSPI XIP configuration
EXT_ADDR	0x20	32	Memory access address register.
TIMING_CFG	0x24	32	HYPERBUS and Octo SPI Memory Timing configuration register.
MBA0	0x28	32	Device start address register.
MBA1	0x2C	32	Device start address register.
DEVICE	0x30	32	Device type register(RAM or FLASH).
OSPI_CMD	0x34	32	OSPI command and psram command config
OSPI_ALTER	0x38	32	OSPI alternative 2 bytes
OSPI_CFG	0x3C	32	OSPI configuration
OSPI_CSN	0x40	32	OSPI chip select configuration
OSPI_JEDEC_RESET	0x44	32	OSPI JEDEC Hardware Reset, user can control sdo0 manually
OSPI_RAM_OPT	0x48	32	OSPI RAM DATA transfer optimisation, only in auto mode
OSPI_ALTER_XIP	0x4C	32	OSPI XIP alternative 2 bytes
OSPI_REG_XIP	0x50	32	OSPI XIP other configuration
LINE_2D	0x54	32	OSPI 2D line.
STRIDE_2D	0x58	32	OSPI 2D stride.
BURST_ENABLE	0x5C	32	OSPI burst mode/2D mode enable.

Name	Offset	Width	Description
IRQ_EN	0x60	32	OSPI interrupt enable register
CLK_DIV	0x64	32	Clock divide.
STATUS	0x68	32	Transfer status for error.
SDIO_CMD_ARG	0x6C	32	SDIO command argument.
SDIO_RSP0	0x70	32	SDIO response 0.
SDIO_RSP1	0x74	32	SDIO response 1.
SDIO_RSP2	0x78	32	SDIO response 2.
SDIO_RSP3	0x7C	32	SDIO response 3.

RX_DEST

Stream ID for the uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the RX 1D/2D uDMA channel. Default is 0xFF(channel disabled)
15:8	R/W	DEST_STREAM	0xFF	Stream ID for the RX STREAM uDMA channel. Default is 0xFF(channel disabled)

TX_DEST

Stream ID for the uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the TX 1D/2D uDMA channel. Default is 0xFF(channel disabled)
15:8	R/W	DEST_STREAM	0xFF	Stream ID for the TX STREAM uDMA channel. Default is 0xFF(channel disabled)

TRANS_MODE

Configure transaction mode

Bit #	R/W	Name	Reset	Description
0	R/W	AUTO_ENA	0x0	Transfer mode in AUTO, IP will configure the UDMA transfer automatically using register parameters instead using SW configuration in UDMA: b0: AUTO_DIS; b1: AUTO_EN
1	R/W	XIP_EN	0x0	Transfer mode in XIP, IP will configure the UDMA transfer automatically using XIP parameters instead using SW configuration in UDMA: b0: XIP_DIS; b1: XIP_EN
3:2	R	RESERVED0	0x0	–
4	R/W	STREAM_EN	0x0	Transfer mode in noraml mode use STREAM or not, IP will configure the STREAM UDMA transfer automatically to read/write data from/to memory: b0: STREAM_DIS; b1: STREAM_EN
5	R/W	AES_STREAM_EN	0x0	Transfer mode in noraml mode use AES STREAM or not, to avoid Read synchronous issue when in AUTO mode: b0: AES_STREAM_DIS; b1: AES_STREAM_EN
7:6	R	RESERVED1	0x0	–
8	R/W	XIP_STREAM_EN	0x0	Transfer mode in noraml mode use STREAM or not, IP will configure the STREAM UDMA transfer automatically to read/write data from/to memory: b0: STREAM_DIS; b1: STREAM_EN
9	R/W	XIP_AES_STREAM_EN	0x0	Transfer mode in noraml mode use AES STREAM or not, to avoid Read synchronous issue when in AUTO mode: b0: AES_STREAM_DIS; b1: AES_STREAM_EN
11:10	R	RESERVED2	0x0	–
12	R/W	XIP_HALTED	0x0	Halted XIP refill when in XIP, XIP refill will wait SW unlock this bit: b0: XIP_RUNNING; b1: XIP_HALTED

TRANS_ADDR

Configure each transaction addr

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDR	0x0	Transfer addr, only when MODE is in AUTO

TRANS_SIZE

Configure each transaction size

Bit #	R/W	Name	Reset	Description
20:0	R/W	SIZE	0x0	Transfer Size

TRANS_CFG

Start each transaction rx/tx

Bit #	R/W	Name	Reset	Description
0	R/W	RXTX	0x0	Transfer type: b0: TX; b1: RX
1	W	VALID	0x0	Transfer valid to start: b0: clear transfer; b1: Start transfer. Read always returns 0.

OSPI_CMD_XIP

OSPI XIP command and psram command config

Bit #	R/W	Name	Reset	Description
15:0	R/W	CMD	0x0	2 Bytes SPI command
22:20	R/W	SDIO_CMD_RSP_-TYPE	0x0	SDIO CMD response type
29:24	R/W	SDIO_CMD_OP	0x0	SDIO CMD operation code

OSPI_CFG_XIP

OSPI XIP configuration

Bit #	R/W	Name	Reset	Description
1:0	R/W	CMD_SIZE	0x0	Octo SPI command size (number of bytes, from 0 to 2)
6:4	R/W	ADDR_SIZE	0x0	Octo SPI address size (number of bytes, from 0 to 4). If 0, jump ADDRESS stage
9:8	R/W	LINE	0x0	Octo SPI number of lines: b00: 8 lines for Octo SPI; b01: 4 lines for QUAD SPI (sdio{3:0]); b10: 1 line for Single SPI (SI: dq[0]; SO: dq[1])
12	R/W	CMD_DTR_STR	0x0	Octo SPI command DDR mode or single mode: b0: DTR mode; b1: STR mode
13	R/W	ADDR_DTR_STR	0x0	Octo SPI address DDR mode or single mode: b0: DTR mode; b1: STR mode
14	R/W	DATA_DTR_STR	0x0	Octo SPI data DDR mode or single mode: b0: DTR mode; b1: STR mode
15	R/W	DATA_DTR_MSB	0x0	Octo SPI data DDR mode data MSB: b0: LSB; b1: MSB

EXT_ADDR

Memory access address register.

Bit #	R/W	Name	Reset	Description
30:0	R/W	SADDR	0x0	Memory access address.
31	R/W	REG_ACCESS	0x0	Register access flag.

TIMING_CFG

HYPERBUS and Octo SPI Memory Timing configuration register.

Bit #	R/W	Name	Reset	Description
4:0	R/W	LATENCY0	0x0	Latency Cycle value for both HyperRAM and HyperFLASH for chip select 0. When using HyperRAM memory, this bit should be set to the same value as the read latency in configuration register of HyperRAM memory. For SPI, it is the number of dummy cycles after ADDRESS stage.
9:5	R/W	LATENCY1	0x0	Latency Cycle value for both HyperRAM and HyperFLASH for chip select 1. When using HyperRAM memory, this bit should be set to the same value as the read latency in configuration register of HyperRAM memory. For SPI, it is the number of dummy cycles after ADDRESS stage.
13:10	R/W	RW_RECOVERY	0x0	Number of clock cycles for Read-Write-Recovery time (tRWR): some HyperBus devices may require a minimum time between the end of a prior transaction and the start of a new access. The master interface waits this number of cycles before driving CS# low after the completion of the CA1 transfer.
16:14	R/W	RWDS_DELAY	0x0	Delay (number of clock cycles) of RWDS for center aligned read.
17	R/W	ADDITIONAL_-LATENCY_AU-TOCHECK_EN	0x0	Autocheck for RWDS high or low for additional latency: b0: no autocheck; b1: autocheck enabled
31:18	R/W	CS_MAX	0x100	Maximum chip select low time for self-refresh HYPERRAM to valid the data transfer (number of clock cycles)

MBA0

Device start address register.

Bit #	R/W	Name	Reset	Description
23:0	R	RESERVED	0x0	–
30:24	R/W	MBA0	0x0	Memory Base Address 0 for both RAM and FLASH bitfield. The base address of addressable region to each memory is set up. Since register can be set in 16M bytes boundary, lower 24 bits are fixed to 0. MBA0 can be greater than MBA1. Used chip select depends on the relationship between MBA0, MBA1 and EXT_ADDR: if MBA0 < MBA1, if MBA1 <= EXT_ADDR then CS1 = 0 else CS0 = 0; if MBA0 > MBA1, if MBA0 <= EXT_ADDR then CS0 = 0 else CS1 = 0.

MBA1

Device start address register.

Bit #	R/W	Name	Reset	Description
23:0	R	RESERVED	0x0	–
30:24	R/W	MBA1	0x0	Memory Base Address for both RAM and FLASH bitfield. The base address of addressable region to each memory is set up. Since register can be set in 16M bytes boundary, lower 24 bits are fixed to 0. MBA0 can be greater than MBA1. Used chip select depends on the relationship between MBA0, MBA1 and EXT_ADDR: if MBA0 < MBA1, if MBA1 <= EXT_ADDR then CSn1 = 0 else CSn0 = 0; if MBA0 > MBA1, if MBA0 <= EXT_ADDR then CSn0 = 0 else CSn1 = 0.

DEVICE

Device type register(RAM or FLASH).

Bit #	R/W	Name	Reset	Description
0	R/W	TYPE	0x0	Device type: b0: Octo/Single SPI; b1: HYPERBUS
1	R/W	DT0	0x0	When in HYPERBUS mode: b0: HYPERRAM; b1: HYPERLASH
2	R/W	DT1	0x0	When in HYPERBUS mode: b0: HYPERRAM; b1: HYPERLASH
3	R/W	SDIO	0x0	SDIO mode

OSPI_CMD

OSPI command and psram command config

Bit #	R/W	Name	Reset	Description
15:0	R/W	CMD	0x0	2 Bytes SPI command
22:20	R/W	SDIO_CMD_RSP_-TYPE	0x0	SDIO CMD response type
29:24	R/W	SDIO_CMD_OP	0x0	SDIO CMD operation code

OSPI_ALTER

OSPI alternative 2 bytes

Bit #	R/W	Name	Reset	Description
15:0	R/W	MODE0	0x0	2 Bytes SPI alternative
31:16	R/W	MODE1	0x0	2 Bytes SPI alternative

OSPI_CFG

OSPI configuration

Bit #	R/W	Name	Reset	Description
1:0	R/W	CMD_SIZE	0x0	Octo SPI command size (number of bytes, from 0 to 2)
6:4	R/W	ADDR_SIZE	0x0	Octo SPI address size (number of bytes, from 0 to 4). If 0, jump ADDRESS stage
9:8	R/W	LINE	0x0	Octo SPI number of lines: b00: 8 lines for Octo SPI; b01: 4 lines for QUAD SPI (sdio{3:0]); b10: 1 line for Single SPI (SI: dq[0]; SO: dq[1])
12	R/W	CMD_DTR_STR	0x0	Octo SPI command DDR mode or single mode: b0: DTR mode; b1: STR mode
13	R/W	ADDR_DTR_STR	0x0	Octo SPI address DDR mode or single mode: b0: DTR mode; b1: STR mode
14	R/W	DATA_DTR_STR	0x0	Octo SPI data DDR mode or single mode: b0: DTR mode; b1: STR mode
15	R/W	DATA_DTR_MSB	0x0	Octo SPI data DDR mode data MSB: b0: LSB; b1: MSB

OSPI_CSN

OSPI chip select configuration

Bit #	R/W	Name	Reset	Description
0	R/W	INDEX	0x0	Octo SPI chip select index controlled by user: b0: CSN0; b1: CSN1
1	R/W	AUTO_EN	0x0	Octo SPI chip select controlled by IP automatically: b0: IP control CSN according to index; b1: IP control CSN according to address range automatically
2	R	SET_CSN_VALID	0x0	Is set to 1 if this register has been configured since the last reset
3	R/W	RESERVED	0x0	–
4	R/W	DIRECT_CTRL	0x0	Octo SPI chip select controlled by user enable GPIO mode: b0: IP control CSN according to index; b1: USER control CSN in GPIO mode
5	R/W	VALUE	0x0	Octo SPI chip select value controlled by user: b0: high; b1: low
6	R/W	SDIO_DATA_QUAD	0x0	SDIO data quad enable: b0: Disable; b1: Enable
7	R/W	SDIO_DATA_QUAD_DDR	0x0	SDIO data quad DDR enable: b0: Disable; b1: Enable
15:8	R/W	SDIO_BLOCK_NUM	0x0	SDIO data block number
25:16	R/W	SDIO_BLOCK_SIZE	0x0	SDIO data block size
26	R/W	SDIO_AUTO_STOP	0x1	SDIO enable HW auto stop after multiple read and write: b0: Disable; b1: Enable

OSPI_JEDEC_RESET

OSPI JEDEC Hardware Reset, user can control sdo0 manually

Bit #	R/W	Name	Reset	Description
0	R/W	USER_CTRL_SDO0_EN	0x0	Octo SPI chip in JEDEC reset mode enable: b0: JEDEC reset disable; b1: USER control sdo0
1	R/W	USER_CTRL_SDO0_VALUE	0x0	Octo SPI chip in JEDEC reset mode, sdo0 value

OSPI_RAM_OPT

OSPI RAM DATA transfer optimisation, only in auto mode

Bit #	R/W	Name	Reset	Description
1:0	R/W	OPT_READ_EN_CS	0x0	Octo SPI RAM optimisation read enable for CS0, special when no accross boundary rwds latency for each channel: b0: disable; b1: enable
3:2	R/W	REAL_ADDR_EN	0x0	Octo SPI or single SPI which use real address instead of address/2 for each channel: b0: disable; b1: enable
5:4	R/W	PSRAM_READ_BIT	0x0	PSRAM CMD automatically reform – CMD bit[47] R/W# – Read is 1 or 0 for each channel
7:6	R/W	PSRAM_CMD_EN	0x0	PSRAM CMD automatically reform enable for each channel
9:8	R/W	PSRAM_ADDR_EVEN	0x0	PSRAM even address reform for each channel
11:10	R/W	PSRAM_CROSS_-BOUNDARY_EN0	0x0	PSRAM cross boundary access (CBD) optimisation R/W enable for channle CS0: b00: Both read write can CBD. b01: read cannot CBD, write can CBD; b10: read can CBD, write cannot CBD; b11: Both read write ca not CBD
13:12	R/W	PSRAM_CROSS_-BOUNDARY_EN1	0x0	PSRAM cross boundary access (CBD) optimisation R/W enable for channle CS0: b00: Both read write can CBD; b01: read cannot CBD, write can CBD; b10: read can CBD, write cannot CBD; b11: Both read write cannot CBD
15:14	R/W	PSRAM_CROSS_-BOUNDARY_PAGE0	0x0	PSRAM cross boundary access optimisation page for channle CS0: b00: 256B; b01: 512B; b10: 1kB; b11: 2kB
17:16	R/W	PSRAM_CROSS_-BOUNDARY_PAGE1	0x0	PSRAM cross boundary access optimisation page for channle CS1: b00: 256B; b01: 512B; b10: 1kB; b11: 2kB

OSPI_ALTER_XIP

OSPI XIP alternative 2 bytes

Bit #	R/W	Name	Reset	Description
15:0	R/W	MODE0	0x0	2 Bytes SPI alternative
31:16	R/W	MODE1	0x0	2 Bytes SPI alternative

OSPI_REG_XIP

OSPI XIP other configuration

Bit #	R/W	Name	Reset	Description
4:0	R/W	XIP_LATENCY0	0x0	XIP latency0 for CS0
9:5	R/W	XIP_LATENCY1	0x0	XIP latency1 for CS1

LINE_2D

OSPI 2D line.

Bit #	R/W	Name	Reset	Description
20:0	R/W	LINE	0x0	OSPI 2D line with 2D mode. For example, ADDR = START_ADDR + i * BURST_STRIDE. Normally, LINE >= BURST_SIZE.

STRIDE_2D

OSPI 2D stride.

Bit #	R/W	Name	Reset	Description
20:0	R/W	STRIDE	0x0	OSPI 2D stride with 2D mode. For example, ADDR = START_ADDR + i * BURST_STRIDE. Normally, STRIDE >= BURST_SIZE.

BURST_ENABLE

OSPI burst mode/2D mode enable.

Bit #	R/W	Name	Reset	Description
0	R/W	CS0_AUTO_BURST_-ENABLE	0x0	Automatically control Maximum chip select low time for self-refresh HYPERRAM to valid the data transfer for channel 0: b0 disable; b1 enable
1	R/W	CS1_AUTO_BURST_-ENABLE	0x0	Automatically control Maximum chip select low time for self-refresh HYPERRAM to valid the data transfer for channel 1: b0 disable; b1 enable
2	R/W	CS0_MAXIMUM_-CHECK_ENABLE	0x0	Enable Maximum chip select low time for self-refresh HYPERRAM for channel 0: b0: disable; b1: enable
3	R/W	CS1_MAXIMUM_-CHECK_ENABLE	0x0	Enable Maximum chip select low time for self-refresh HYPERRAM for channel 1: b0: disable; b1: enable
5:4	R/W	QSPI_2D_ENABLE	0x0	OSPI burst 2D mode enable for normal mode and XIP: b0: disable; b1: enable
7:6	R/W	L2_2D_MODE	0x0	2D tansfer mode from L2 to external memory config: b00: 1D_TO_1D; b01: 1D_TO_2D; b10: 2D_TO_1D; b11: 2D_TO_2D

IRQ_EN

OSPI interrupt enable register

Bit #	R/W	Name	Reset	Description
0	R/W	EN	0x0	Octo SPI interrupt enable control: b0: interrupt disable; b1: Interrupt enable
1	R/W	XIP_EN	0x0	Octo SPI interrupt enable control for XIP: b0: interrupt disable; b1: Interrupt enable

CLK_DIV

Clock divide.

Bit #	R/W	Name	Reset	Description
7:0	R/W	DATA	0x0	Clock divider ratio (0-255): PERIPH clock frequency is divided by DATA

STATUS

Transfer status for error.

Bit #	R/W	Name	Reset	Description
0	R	TX_ERROR	0x0	TX transfer error because of tcsm, write 1 to clear: b0: no error; b1: error
1	R	RX_ERROR	0x0	RX transfer error because of tcsm, write 1 to clear: b0: no error; b1: error
2	R	RX_TX_END	0x0	RX TX transfer end flag, can be polled by user, write 1 to clear: b0: not end; b1: end
3	R	SDIO_RX_TX_ERROR	0x0	SDIO RX TX transfer error because of tcsm, write 1 to clear: b0: no error; b1: error
4	R	SDIO_RX_TX_END	0x0	SDIO RX TX transfer end flag, can be polled by user, write 1 to clear: b0: not end; b1: end
15:5	R	RESERVED	0x0	–
31:16	R	SDIO_ERROR_STATUS	0x0	SDIO error status flag, indicate the error type

SDIO_CMD_ARG

SDIO command argument.

Bit #	R/W	Name	Reset	Description
31:0	R/W	ARG	0x0	SDIO command argument

SDIO_RSP0

SDIO response 0.

Bit #	R/W	Name	Reset	Description
31:0	R	RSP0	0x0	SDIO response 0

SDIO_RSP1

SDIO response 1.

Bit #	R/W	Name	Reset	Description
31:0	R	RSP1	0x0	SDIO response 1

SDIO_RSP2

SDIO response 2.

Bit #	R/W	Name	Reset	Description
31:0	R	RSP2	0x0	SDIO response 2

SDIO_RSP3

SDIO response 3.

Bit #	R/W	Name	Reset	Description
31:0	R	RSP3	0x0	SDIO response 3

5.3.6 Serial Audio Interface

The SAI interface supports the following features:

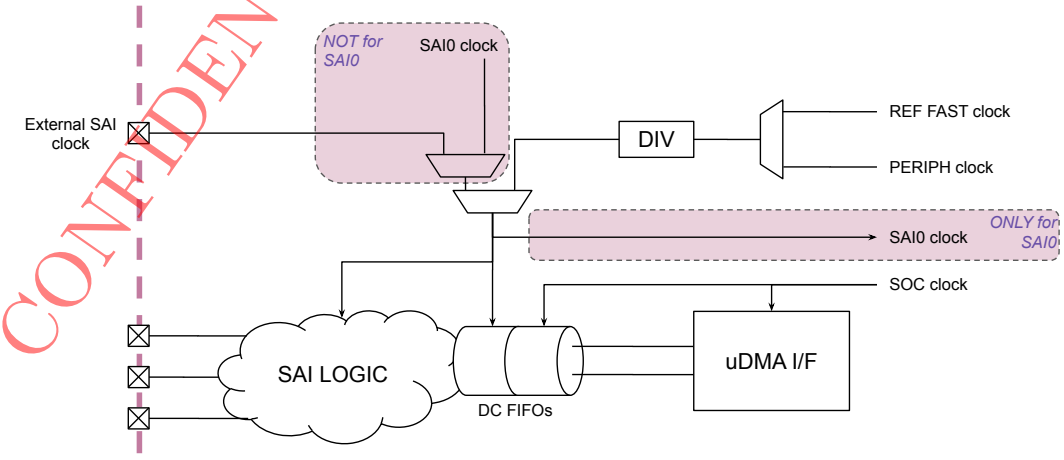
- Support of legacy I2S interface.
- Clock and strobe (WS) internally generated or from external component.
- Configurable clock, WS and data polarities.
- Data size up to 32 bits.
- Half and full duplex modes (single bidirectional lane or separate SDI/SDO lanes).
- Up to 16 TDM slots with independent data size, bit order and justification configuration.
- PDM mode (2 slave channels or 1 master channel per lane).
- Optional synchronized start of serial audio interfaces.

5.3.6.1 Clocking

The figure below shows the clocking scheme of the SAI peripheral. The clock may be chosen between the following sources:

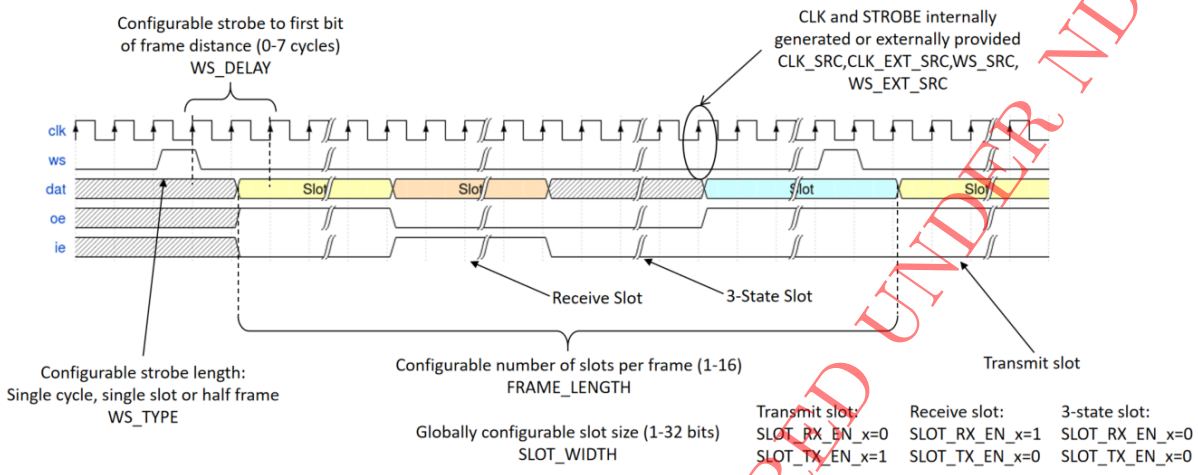
- PERIPH clock generated by the FLL, on which an integer division may be applied,
- REF clock, on which an integer division may be applied,
- for SAI1 and SAI2 only, the clock used for SAI0,
- the external SAI clock, when the interface is slave from the clock and start of frame (WS) signals perspective.

Note that the SAI configuration interface – not shown on the figure – is clocked by SOC clock.

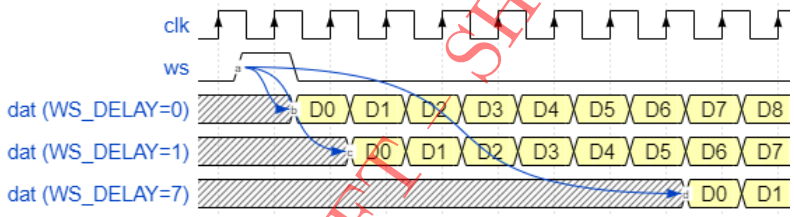


5.3.6.2 Timing Diagrams

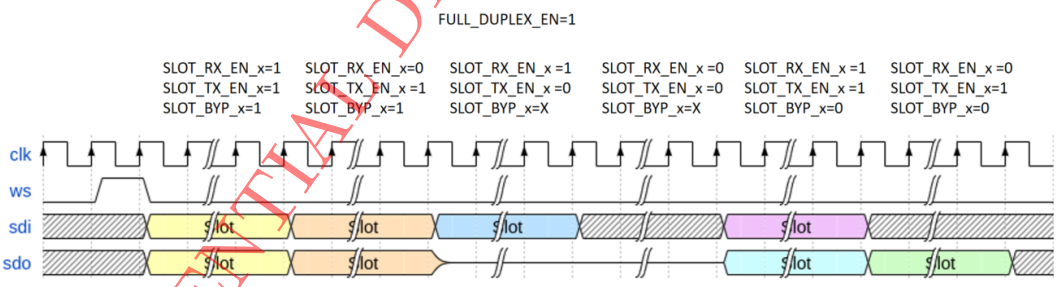
This section gives more details on the SAI configuration possibilities and their impact on the interface signals. The figure below sums up the possible configurations for frame format, with the names of the corresponding register fields.



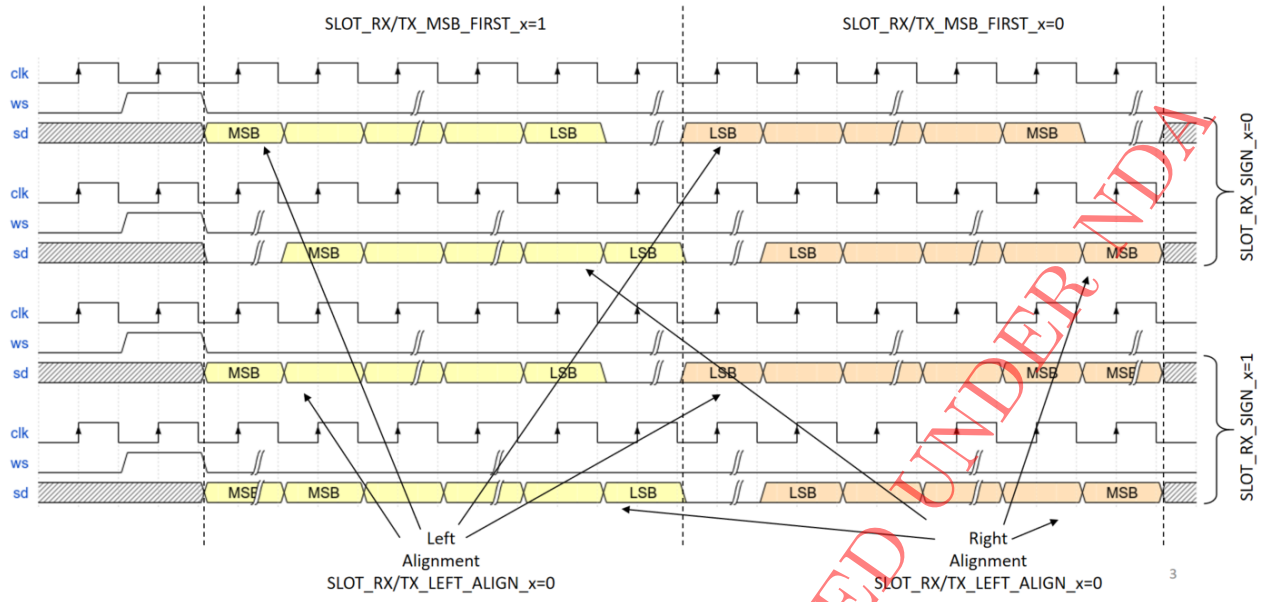
The WS_DELAY field of the GLB_SETUP register configures the delay, in number of clock cycles, between the frame strobe (WS signal) and the first bit of the first slot of the frame.



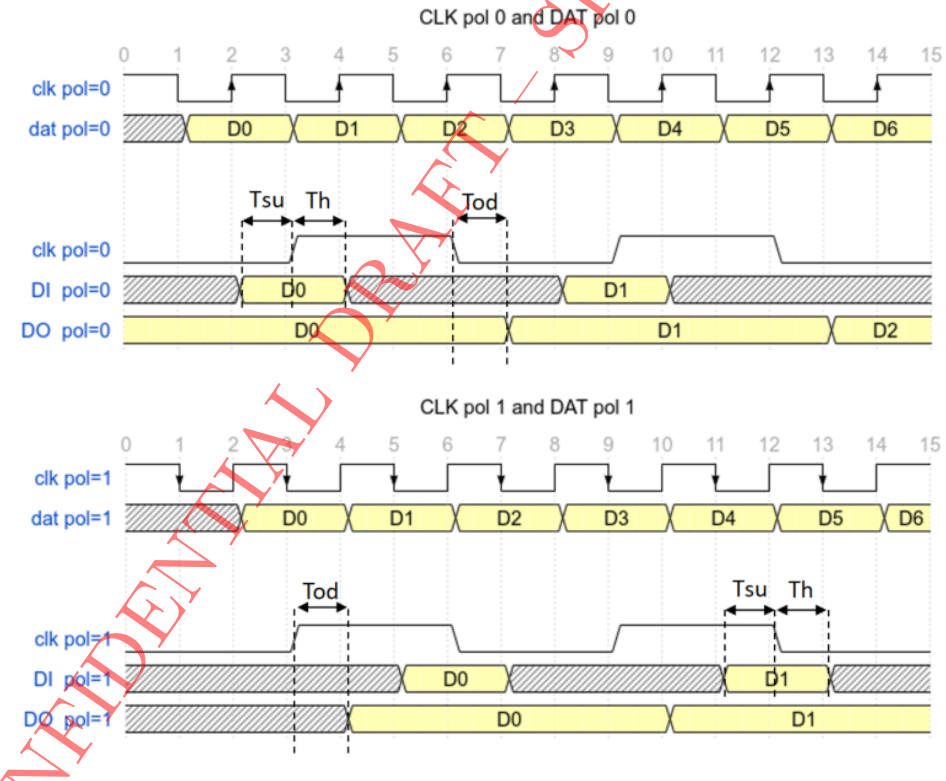
In half duplex mode, SDI pad is used for both data input and data output, and is left in high-impedance state if the slot is not used. In full duplex mode, both SDI and SDO are used, and their behavior is shown on the next figure.

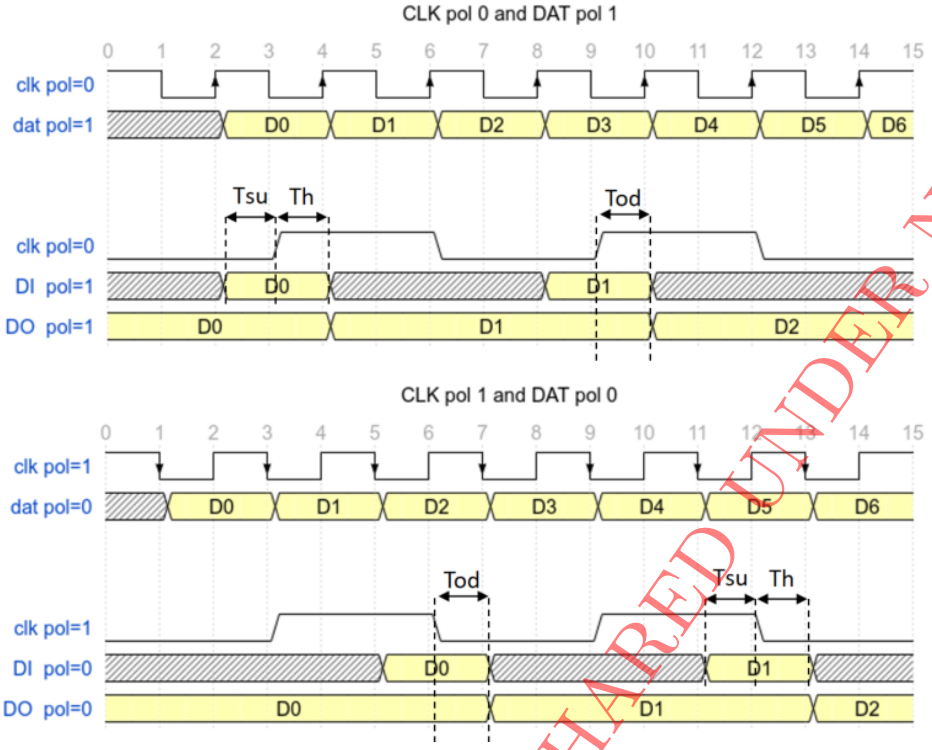


The next figure shows the different data alignement possibilities according to the settings in SLOT_CFG_* registers. When data size for a slot is lower than the global slot size setting, extra bits are inserted, also taking into account the signed extension configuration.



The next four figures illustrate the impact of the settings of clock and data polarities. `Tsu`, `Th` and `Tod` refer to the setup, hold and output delay times, respectively. `Tsu` and `Th` show the constraint of input signal stability around the clock edge used to sample the value. `Tod` shows the delay to guarantee a stable output signal after the clock edge.





Logic voltage	Source clock		Normal mode, SAI0 and SAI1			Normal mode, SAI2			Bypass mode, SAI0, SAI1 and SAI2		
			Tsu	Th	Tod	Tsu	Th	Tod	Tsu	Th	Tod
0.8V	Internal	REF FAST clock	6.0ns	6.0ns	6.0ns	6.0ns	6.0ns	6.0ns	8.0ns	8.0ns	8.0ns
		PERIPH clock	5.7ns	5.7ns	5.7ns	5.7ns	5.7ns	5.7ns	7.5ns	7.5ns	7.5ns
	External clock from SAI pad		6.2ns	6.2ns	6.2ns	6.2ns	6.2ns	6.2ns	8.0ns	8.0ns	8.0ns
0.65V	Internal	REF FAST clock	6.0ns	6.0ns	6.0ns	6.6ns	6.6ns	6.6ns	8.0ns	8.0ns	8.0ns
		PERIPH clock	6.2ns	6.2ns	6.2ns	6.9ns	6.9ns	6.9ns	8.4ns	8.4ns	8.4ns
	External clock from SAI pad		8.3ns	8.3ns	8.3ns	9.0ns	9.0ns	9.0ns	10.5ns	10.5ns	10.5ns

5.3.6.3 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
SLOT_CFG_0	0x0	32	Configuration for slot 0
SLOT_CFG_1	0x4	32	Configuration for slot 1
SLOT_CFG_2	0x8	32	Configuration for slot 2
SLOT_CFG_3	0xC	32	Configuration for slot 3
SLOT_CFG_4	0x10	32	Configuration for slot 4
SLOT_CFG_5	0x14	32	Configuration for slot 5
SLOT_CFG_6	0x18	32	Configuration for slot 6
SLOT_CFG_7	0x1C	32	Configuration for slot 7
SLOT_CFG_8	0x20	32	Configuration for slot 8
SLOT_CFG_9	0x24	32	Configuration for slot 9
SLOT_CFG_10	0x28	32	Configuration for slot 10
SLOT_CFG_11	0x2C	32	Configuration for slot 11
SLOT_CFG_12	0x30	32	Configuration for slot 12
SLOT_CFG_13	0x34	32	Configuration for slot 13
SLOT_CFG_14	0x38	32	Configuration for slot 14
SLOT_CFG_15	0x3C	32	Configuration for slot 15
WORD_SIZE_0_1	0x40	32	Defines word size for RX and TX channels for slot 0 and 1
WORD_SIZE_2_3	0x44	32	Defines word size for RX and TX channels for slot 2 and 3
WORD_SIZE_4_5	0x48	32	Defines word size for RX and TX channels for slot 4 and 5
WORD_SIZE_6_7	0x4C	32	Defines word size for RX and TX channels for slot 6 and 7
WORD_SIZE_8_9	0x50	32	Defines word size for RX and TX channels for slot 8 and 9
WORD_SIZE_10_11	0x54	32	Defines word size for RX and TX channels for slot 10 and 11
WORD_SIZE_12_13	0x58	32	Defines word size for RX and TX channels for slot 12 and 13
WORD_SIZE_14_15	0x5C	32	Defines word size for RX and TX channels for slot 14 and 15
SLOT_EN	0x60	32	Slot Enable
CLKCFG_SETUP	0x64	32	Clock configuration for both master, slave and PDM
GLB_SETUP	0x68	32	Configuration of the global parameters
CLK_FAST	0x6C	32	Configuration to use reference fast clock
ERR_STATUS	0x70	32	Error status for all slots 0-15 RX slots 16-31 TX slots

SLOT_CFG_0

Configuration for slot 0

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_1

Configuration for slot 1

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_2

Configuration for slot 2

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_3

Configuration for slot 3

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_4

Configuration for slot 4

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_5

Configuration for slot 5

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_6

Configuration for slot 6

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_7

Configuration for slot 7

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_8

Configuration for slot 8

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_9

Configuration for slot 9

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_10

Configuration for slot 10

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_11

Configuration for slot 11

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_12

Configuration for slot 12

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_13

Configuration for slot 13

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_14

Configuration for slot 14

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

SLOT_CFG_15

Configuration for slot 15

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_ID	0xFF	UDMA stream ID for RX channel
8	R/W	RX_EN	0x0	Enable RX for the slot
9	R/W	RX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
10	R/W	RX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right ; b1: left
13:12	R/W	RX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
14	R/W	RX_SIGN	0x0	Enables sign extension on RX data
23:16	R/W	TX_ID	0xFF	UDMA stream ID for TX channel
24	R/W	TX_EN	0x0	Enable TX for the slot
25	R/W	TX_MSB_FIRST	0x0	Configuration of the MSB or LSB first: b0: LSB; b1: MSB
26	R/W	TX_LEFT_ALIGN	0x0	Justification to the left or to the right: b0: right; b1: left
29:28	R/W	TX_DSIZE	0x0	UDMA transfer size: b00: 1 byte; b01: 2 bytes; b10: 3 bytes; b11: 4 bytes
30	R/W	TX_SIGN	0x0	Enables sign extension on TX data
31	R/W	BYP	0x0	Enables input-to-output bypass: when set to b1, the slot input value is directly redirected to the output, in the same slot

WORD_SIZE_0_1

Defines word size for RX and TX channels for slot 0 and 1

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_2_3

Defines word size for RX and TX channels for slot 2 and 3

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_4_5

Defines word size for RX and TX channels for slot 4 and 5

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_6_7

Defines word size for RX and TX channels for slot 6 and 7

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_8_9

Defines word size for RX and TX channels for slot 8 and 9

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_10_11

Defines word size for RX and TX channels for slot 10 and 11

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_12_13

Defines word size for RX and TX channels for slot 12 and 13

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

WORD_SIZE_14_15

Defines word size for RX and TX channels for slot 14 and 15

Bit #	R/W	Name	Reset	Description
4:0	R/W	WORD_SIZE_RX_0	0x0	Word size for RX channel of slots 0
12:8	R/W	WORD_SIZE_TX_0	0x0	Word size for TX channel of slots 0
20:16	R/W	WORD_SIZE_RX_1	0x0	Word size for RX channel of slots 1
28:24	R/W	WORD_SIZE_TX_1	0x0	Word size for TX channel of slots 1

SLOT_EN

Slot Enable

Bit #	R/W	Name	Reset	Description
0	R/W	SLOT_EN_0	0x0	Enable for slot 0
1	R/W	SLOT_EN_1	0x0	Enable for slot 1
2	R/W	SLOT_EN_2	0x0	Enable for slot 2
3	R/W	SLOT_EN_3	0x0	Enable for slot 3
4	R/W	SLOT_EN_4	0x0	Enable for slot 4
5	R/W	SLOT_EN_5	0x0	Enable for slot 5
6	R/W	SLOT_EN_6	0x0	Enable for slot 6
7	R/W	SLOT_EN_7	0x0	Enable for slot 7
8	R/W	SLOT_EN_8	0x0	Enable for slot 8
9	R/W	SLOT_EN_9	0x0	Enable for slot 9
10	R/W	SLOT_EN_10	0x0	Enable for slot 10
11	R/W	SLOT_EN_11	0x0	Enable for slot 11
12	R/W	SLOT_EN_12	0x0	Enable for slot 12
13	R/W	SLOT_EN_13	0x0	Enable for slot 13
14	R/W	SLOT_EN_14	0x0	Enable for slot 14
15	R/W	SLOT_EN_15	0x0	Enable for slot 15

CLKCFG_SETUP

Clock configuration for both master, slave and PDM

Bit #	R/W	Name	Reset	Description
15:0	R/W	CLK_DIV	0x0	Clock divider
16	R/W	CLK_EN	0x0	Enables clock generator
17	R/W	CLK_SRC	0x0	Clock internal or external: b0: internal; b1: external
18	R/W	CLK_EXT_SRC	0x0	Clock external source: b0: pad; b1: internally routed (from SAI0)
19	R/W	CLK_EDGE	0x0	Clock polarity: b0: rising edge; b1: falling edge
20	R/W	WS_SRC	0x0	WS internal or external: b0: internal; b1: external
21	R/W	WS_EXT_SRC	0x0	WS external source: b0: pad; b1: internally routed (from SAI0)
22	R/W	WS_EDGE	0x0	WS polarity: b0: rising edge; b1: falling edge
24:23	R/W	WS_TYPE	0x0	WS type (duration): b00: pulse (first bit of first slot); b01: first slot; b10: half frame (N/2 slots); b11: pulse

GLB_SETUP

Configuration of the global parameters

Bit #	R/W	Name	Reset	Description
0	R/W	GLOBAL_EN	0x0	Enables the I2S interface
4:1	R/W	FRAME_LENGTH	0x0	Sets how many slots for each frame (1-16)
9:5	R/W	SLOT_WIDTH	0x0	Sets the slot width in bits (1-32)
12:10	R/W	WS_DELAY	0x0	Sets the distance in I2S cycles from the WS rising edge to the first bit of the frame
13	R/W	FULL_DUPLEX_EN	0x0	Enables full duplex mode (SDI and SDO)
14	R/W	PDM_EN	0x0	Switch to PDM mode on SDI (bit0) and SDO (bit1) (2 PDM lanes, 2 slave channels or 1 master channel per lane)
16:15	R/W	PDM_POL	0x0	Set lane polarity (0: slave, 1: master) for SDI (bit0) and SDO (bit1)
18:17	R/W	PDM_DIFF	0x0	In master mode only: set differential mode on pairs, bit0: (SDI,WS), bit1: (SDO,SCK)
21:19	R/W	BLOCK_COMMIT	0x0	Used for synchronization of the 3 SAI interfaces: GLOBAL_EN and SLOT_EN_* fields are taken into account on SAIi only when writing b1 into bit i of this field. This write can be done through any of the SAI interfaces, to have several interfaces start exactly at the same time

CLK_FAST

Configuration to use reference fast clock

Bit #	R/W	Name	Reset	Description
0	R/W	FAST_EN	0x0	Enables to use reference fast clock

ERR_STATUS

Error status for all slots 0-15 RX slots 16-31 TX slots

Bit #	R/W	Name	Reset	Description
15:0	R/W	RX_ERR_FLAG	0x0	Error status flag for RX slots: bit i=1 signals an error on RX slot i
31:16	R/W	TX_ERR_FLAG	0x0	Error status flag for TX slots: bit i=1 signals an error on TX slot i

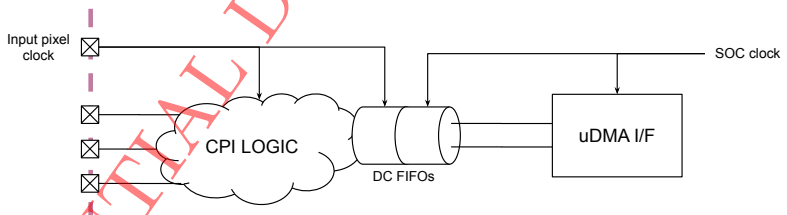
5.3.7 CPI Interface

Control of the 8-bit camera parallel interface.

Note: CPI interface I/Os are not available in the WLCSP package.

5.3.7.1 Clocking

The figure below shows the clocking scheme of the CPI peripheral. The peripheral is directly clocked by the *pixel clock* which is transmitted by the external image sensor. Note that the CPI configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.7.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CAM_RX_DEST	0x0	32	Stream ID for the uDMA channel
CAM_RX_DATASIZE	0x4	32	Data size
CAM_CFG_GLOB	0x20	32	Global configuration register
CAM_CFG_LL	0x24	32	Lower Left corner configuration register
CAM_CFG_UR	0x28	32	Upper Right corner configuration register
CAM_CFG_SIZE	0x2C	32	Horizontal Resolution configuration register
CAM_CFG_RGB	0x30	32	RGB coefficients configuration register
CAM_VSYNC_POLARITY	0x34	32	VSYNC Polarity register

CAM_RX_DEST

Stream ID for the uDMA channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_DEST	0xFF	Stream ID for the uDMA channel. Default is 0xFF (channel disabled).

CAM_RX_DATASIZE

Data size

Bit #	R/W	Name	Reset	Description
1:0	R/W	RX_DATASIZE	0x0	Transfer datasize: b00: 8bit; b01: 16bit; b10: 24bit; b11: 32bit

CAM_CFG_GLOB

Global configuration register

Bit #	R/W	Name	Reset	Description
0	R/W	FRAMEDROP_EN	0x0	Frame dropping: b0: disabled; b1: enabled
6:1	R/W	FRAMEDROP_VAL	0x0	Set how many frames should be dropped after each received
7	R/W	FRAMESLICE_EN	0x0	Input frame slicing: b0: disabled; b1: enabled
10:8	R/W	FORMAT	0x0	Input frame format: b000: RGB565; b001: RGB555; b010: RGB444; b011: RGB888; b100: BYPASS_LITEND; b101: BYPASS_BIGEND
31	R/W	EN	0x0	Enable data rx from camera interface. The enable/disable happens only at the start of a frame. b0: disabled; b1: enabled

CAM_CFG_LL

Lower Left corner configuration register

Bit #	R/W	Name	Reset	Description
15:0	R/W	FRAMESLICE_LLX	0x0	X coordinate of lower left corner of slice
31:16	R/W	FRAMESLICE_LLY	0x0	Y coordinate of lower left corner of slice

CAM_CFG_UR

Upper Right corner configuration register

Bit #	R/W	Name	Reset	Description
15:0	R/W	FRAMESLICE_URX	0x0	X coordinate of upper right corner of slice
31:16	R/W	FRAMESLICE_URY	0x0	Y coordinate of upper right corner of slice

CAM_CFG_SIZE

Horizontal Resolution configuration register

Bit #	R/W	Name	Reset	Description
15:0	R/W	ROWLEN	0x0	Horizontal Resolution. It is used for slice mode. Value set into the bitfield must be equal to (rowlen-1).

CAM_CFG_RGB

RGB coefficients configuration register

Bit #	R/W	Name	Reset	Description
2:0	R/W	FORMAT	0x0	Order of RGB coefficients: h0: RGB; h1: RBG; h2: GRB; h3: GBR; h4: BRG; h5: BGR; h6: RGB; h7: RGB

CAM_VSYNC_POLARITY

VSYNC Polarity register

Bit #	R/W	Name	Reset	Description
0	R/W	VSYNC_POLARITY	0x0	Set vsync polarity of CPI: b0: active low; b1: active high
1	R/W	HSYNC_POLARITY	0x0	Set hsync polarity of CPI: b0: active high; b1: active low

5.3.8 CSI-2 Interface

The CSI-2 interface of GAP9 offers 1 clock lane and 2 data lanes (only 1 data lane is available in WLCSP package), and supports transfer rates up to 1.5Gbps per data lane. It is actually made up of three parts: a digital PHY, compliant with the MIPI® Alliance D-PHY Specification v1.2, a controller in charge of the CSI-2 protocol, compliant with the MIPI® Alliance CSI-2 Specification v1.3, and a block responsible for interfacing with GAP9 uDMA, that aligns the pixels to 8-bit boundaries of 32-bit words and is capable of clipping the picture to only retrieve a rectangular region of interest. Each of these 3 parts has its specific memory map access range, as described below.

The CSI-2 comes with a Camera Control Interface (CCI), which is basically an I2C interface dedicated to configuring the distant camera. It is available on the reconfigurable pads of the chip.

5.3.8.1 Clocking

The figure below shows the clocking scheme of the CSI-2 RX peripheral, which uses three main clocks: the *PHY byte clock*, the *lane clock* and the *pixel clock*.

The *PHY byte clock* is automatically derived from the external differential clock coming from the external CSI-2 sensor. This clock is usually configurable in the sensor, according to the image size and frame rate, using the CCI interface. The differential clock, which is used in DDR mode as a bit clock, is converted into a byte clock by the D-PHY block. For example, when the CSI-2 lane datarate is 1.5Gbps, the differential clock is 750MHz DDR, and the PHY byte clock is 1500/8=187.5MHz, meaning that each lane delivers 187,500 bytes per second.

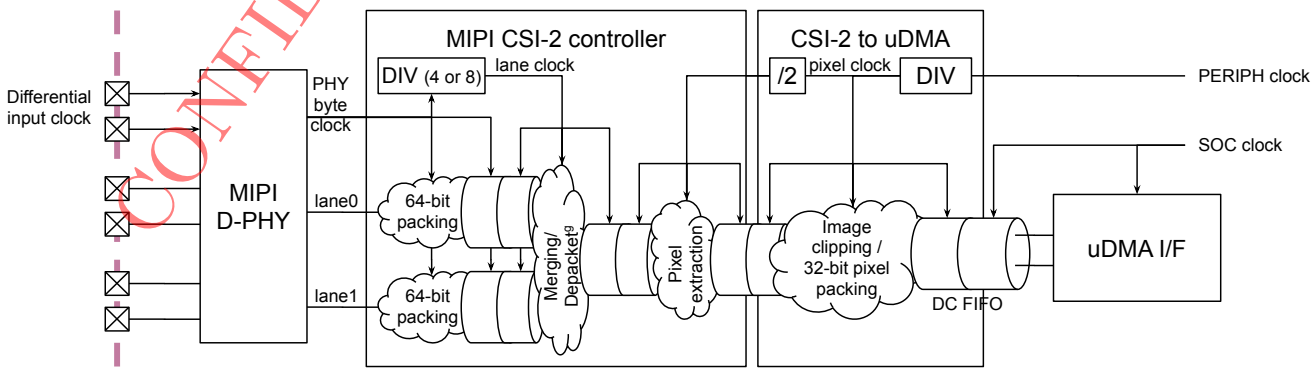
Bytes from the different lanes are then gathered into 64-bit words in the CSI-2 RX controller internal buffer. The *lane clock* is therefore a divided verion of the PHY byte clock, divided by 4 if 2 lanes are active, or by 8 if a single lane is active.

The *pixel clock* is derived from the PERIPH clock generated by the FLL, on which a configurable integer division is applied. It is used to retrieve the pixels from the 64-bit buffer (at half rate), apply an optional clipping to select a region of interest, then send the pixels into a 32-bit FIFO, from which they can be retrieved by the uDMA.

Note: Lane clock and pixel clock frequencies must be carefully configured, so that pixels are read at a correct rate.

The pixel clock should be set so match the frequency required to read the pixels received from the CSI-2 interface. For example, for a single lane at 1.5Gbps, taking into account that the useful payload of MIPI transfers is 90%, there is 168,750 bytes of pixel data received per second. If the pixel format is RAW10, this means 135,000 pixels per second. A 32-bit word read from the controller contains 2 packed pixels. As a consequence the pixel clock should be set to 67.5MHz. If the pixel format is RAW8, there are 168,750 pixels received per second. A 32-bit word contains 4 pixels, therefore the pixel clock should be set to 42.2MHz. If the pixel format is RGB888, there are 56,250 pixels received per second. A 32-bit word contains a single pixel, so the pixel clock should be set to 56.3MHz.

Also note that the CSI-2 configuration interfaces – not shown on the figure – are clocked by SOC clock.



5.3.8.2 Register map for CSI-2 UDMA interface

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_CH0_DEST	0x0	32	Stream ID for the uDMA channel 0
RX_CH1_DEST	0x4	32	Stream ID for the uDMA channel 1
RX_ENABLE	0x20	32	Enable receiving
CLK_SETUP	0x24	32	Configure clock divider
SLICE_ENABLE	0x28	32	Enable slice mode
SLICE_ROWLEN	0x2C	32	Configure image row length when in slice mode
SLICE0_ULXY	0x30	32	Configure upper left pixel position when in slice mode for channel 0
SLICE0_BRXY	0x34	32	Configure bottom right pixel position when in slice mode for channel 0
SLICE1_ULXY	0x38	32	Configure upper left pixel position when in slice mode for channel 1
SLICE1_BRXY	0x3C	32	Configure bottom right pixel position when in slice mode for channel 1

RX_CH0_DEST

Stream ID for the uDMA channel 0

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the CMD uDMA channel 0. Default is 0xFF (channel disabled)

RX_CH1_DEST

Stream ID for the uDMA channel 1

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the CMD uDMA channel 1. Default is 0xFF (channel disabled)

RX_ENABLE

Enable receiving

Bit #	R/W	Name	Reset	Description
0	R/W	EN0	0x0	Enable channel 0 pixel stream receiving: 0: Disable, 1: Enable
1	R/W	EN1	0x0	Enable channel 1 pixel stream receiving: 0: Disable, 1: Enable

CLK_SETUP

Configure clock divider

Bit #	R/W	Name	Reset	Description
7:0	R/W	CCI_CLK_DIV	0x0	Clock diver for CCI clock
15:8	R/W	PIXEL_CLK_DIV	0x0	Clock diver for pixel clock
23:16	R/W	APB_CLK_DIV	0x0	Clock diver for APB clock

SLICE_ENABLE

Enable slice mode

Bit #	R/W	Name	Reset	Description
0	R/W	EN0	0x0	Enable channel 0 slice mode for pixel stream receiving: 0: Disable, 1: Enable
1	R/W	EN1	0x0	Enable channel 1 slice mode for pixel stream receiving: 0: Disable, 1: Enable

SLICE_ROWLEN

Configure image row length when in slice mode

Bit #	R/W	Name	Reset	Description
15:0	R/W	ROWLEN0	0x0	Slice mode image row length for channel 0
31:16	R/W	ROWLEN1	0x0	Slice mode image row length for channel 1

SLICE0_ULXY

Configure upper left pixel position when in slice mode for channel 0

Bit #	R/W	Name	Reset	Description
15:0	R/W	ULY	0x0	Upper left pixel position Y for channel 0
31:16	R/W	ULX	0x0	Upper left pixel position X for channel 0

SLICE0_BRXY

Configure bottom right pixel position when in slice mode for channel 0

Bit #	R/W	Name	Reset	Description
15:0	R/W	BRY	0x0	Bottom right pixel position Y for channel 0
31:16	R/W	BRX	0x0	Bottom right pixel position X for channel 0

SLICE1_ULXY

Configure upper left pixel position when in slice mode for channel 1

Bit #	R/W	Name	Reset	Description
15:0	R/W	ULY	0x0	Upper left pixel position Y for channel 1
31:16	R/W	ULX	0x0	Upper left pixel position X for channel 1

SLICE1_BRXY

Configure bottom right pixel position when in slice mode for channel 1

Bit #	R/W	Name	Reset	Description
15:0	R/W	BRY	0x0	Bottom right pixel position Y for channel 1
31:16	R/W	BRX	0x0	Bottom right pixel position X for channel 1

5.3.8.3 Register map for MIPI CSI-2 Controller

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CONFIG	0x180	32	Configuration of the number of active lanes
ERR_MSB1	0x184	32	Error status MSB1
ERR_MSB	0x188	32	Error status MSB
ERR_LSB	0x18C	32	Error status LSB
HS_RX_TIMEOUT_-MSB2	0x190	32	Configuration of timeout in high speed mode (MSB2)
HS_RX_TIMEOUT_-MSB1	0x194	32	Configuration of timeout in high speed mode (MSB1)
HS_RX_TIMEOUT_-LSB	0x198	32	Configuration of timeout in high speed mode (LSB)
VCCFG	0x19C	32	Virtual Channel configuration
POLARITY	0x1A0	32	Vsync and Hsync polarity
CCI_ADDRESS	0x1A4	32	CCI address
CCI_WRITE_DATA	0x1A8	32	CCI write data
CCI_READ_DATA	0x1AC	32	CCI read data
CCI_READ_WRITE	0x1B0	32	CCI R/W
CCI_STATUS	0x1B4	32	CCI status
CCI_DEV_ADDR	0x1B8	32	CCI device ID
ULPS_STATUS	0x1BC	32	CSI-2 ULPS status

CONFIG

Configuration of the number of active lanes

Bit #	R/W	Name	Reset	Description
1:0	R/W	CSI_CONFIG	0x00	Number of active lanes: b00: single lane, b01: two lanes, b11: four lanes

ERR_MSB1

Error status MSB1

Bit #	R/W	Name	Reset	Description
0	R	CRC_ERROR_VC_3	0x0	Set to 1 if there is a checksum error on virtual channel 3 (Checksum Error Long packet only)
1	R	ERR_ESC	0x0	Set to 1 if there is an error in escape entry command

ERR_MSB

Error status MSB

Bit #	R/W	Name	Reset	Description
0	R	INVLD_PKT_LEN	0x0	Set to 1 if there is an invalid packet length (invalid transmission length)
1	R	FRAME_SYNC_ERR	0x0	Set to 1 if a frame end is received but not paired with a frame start in the same virtual channel
2	R	ECC_NO_ERR	0x0	Set to 1 when ECC check shows no error (either no error or more than 2 bits of error)
3	R	ECC_BIT_ERROR	0x0	Set to 1 if there is an error in the ECC field
4	R	ERR_FRAME_DATA	0x0	If a CRC error is present in the data packet, then this error is set to 1 when vsync end is received
5	R	HS_RX_TO_ERR	0x0	Set to 1 in case of HS RX timeout
6	R	CRC_ERROR_VC1	0x0	Set to 1 if there is a checksum error on virtual channel 1 (Checksum Error Long packet only)
7	R	CRC_ERROR_VC2	0x0	Set to 1 if there is a checksum error on virtual channel 2 (Checksum Error Long packet only)

ERR_LSB

Error status LSB

Bit #	R/W	Name	Reset	Description
0	R	SOT_ERR	0x0	Set to 1 if there is an error with start of frame (SoT Error)
1	R	SOT_SYNC_ERR	0x0	Set to 1 if there is an error in synchronization of Start of Transfer (SoT Sync Error)
2	R	FALSE_CTRL	0x0	Set to 1 if there is a False Control Error
3	R	ECC_ERR_SINGLE	0x0	Set to 1 if there is a single bit error, even when it is corrected using ECC (ECC Error, single-bit detected and corrected)
4	R	ECC_ERR_MULT	0x0	Set to 1 if there is a two-bit error in the packet (ECC Error, multi-bit detected not corrected)
5	R	CRC_ERROR_VC0	0x0	Set to 1 if there is a checksum error on virtual channel 0 (Checksum Error Long packet only)
6	R	INVLD_DATA_TYPE	0x0	Set to 1 if the received data is invalid (CSI Data Type Not Recognized)
7	R	INVLD_VC_ID	0x0	Set to 1 in case of invalid virtual channel ID (CSI VC ID Invalid)

HS_RX_TIMEOUT_MSB2

Configuration of timeout in high speed mode (MSB2)

Bit #	R/W	Name	Reset	Description
7:0	R/W	TIME_OUT	0xFF	High speed request timeout configuration (bits 16 to 23)

HS_RX_TIMEOUT_MSB1

Configuration of timeout in high speed mode (MSB1)

Bit #	R/W	Name	Reset	Description
7:0	R/W	TIME_OUT	0xFF	High speed request timeout configuration (bits 8 to 15)

HS_RX_TIMEOUT_LSB

Configuration of timeout in high speed mode (LSB)

Bit #	R/W	Name	Reset	Description
7:0	R/W	TIME_OUT	0xFF	High speed request timeout configuration (bits 0 to 7)

VCCFG

Virtual Channel configuration

Bit #	R/W	Name	Reset	Description
0	R/W	VCCFG	0x1	Set bit to 1 to enable virtual channel (default: VC0 enabled)

POLARITY

Vsync and Hsync polarity

Bit #	R/W	Name	Reset	Description
0	R/W	VSYNC	0x0	VSYNC polarity: 0: active high, 1: active low
1	R/W	HSYNC	0x0	HSYNC polarity: 0: active high, 1: active low

CCI_ADDRESS

CCI address

Bit #	R/W	Name	Reset	Description
7:0	R/W	ADDRESS	0x00	CCI interface address

CCI_WRITE_DATA

CCI write data

Bit #	R/W	Name	Reset	Description
7:0	R/W	WR_DATA	0x00	CCI interface write data

CCI_READ_DATA

CCI read data

Bit #	R/W	Name	Reset	Description
7:0	R	RD_DATA	0x00	CCI interface read data

CCI_READ_WRITE

CCI R/W

Bit #	R/W	Name	Reset	Description
6:0	R/W	CCI_BYTE	0x00	CCI byte
7	W	CCI_WRITE	0x0	CCI write

CCI_STATUS

CCI status

Bit #	R/W	Name	Reset	Description
0	R/W	RW_DONE	0x0	Read/write transfer done (write 1 to clear)
1	R	READ_READ	0x0	Is set to 1 if CCI read data available

CCI_DEV_ADDR

CCI device ID

Bit #	R/W	Name	Reset	Description
6:0	R/W	ADDR	0x3C	CCI device address

ULPS_STATUS

CSI-2 ULPS status

Bit #	R/W	Name	Reset	Description
0	R	ULPS_ACTIVE_-LANE0	0x0	Set to 1 if ultra low power state is active for data lane 0
1	R	ULPS_ACTIVE_-LANE1	0x0	Set to 1 if ultra low power state is active for data lane 1
4	R	ULPS_ACTIVE_CLK	0x0	Set to 1 if ultra low power state is active for clock lane

5.3.8.4 Register map for MIPI Digital PHY

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
REG00	0x0	32	Data lane and clock lane enable register
REG0D	0x34	32	Digital clock sample manual phase selection register
REG0E	0x38	32	Clock lane manual phase selection register
REG0F	0x3C	32	Data lanes manual phase selection register
REG12	0x48	32	Reverse digital sample clock register
REG20	0x80	32	Digital reset register
REG4A	0x128	32	Clock lane continuous mode register
REG58	0x160	32	Clock lane TSH_SETTLE register
REG5A	0x168	32	Clock lane calibration reception register
REG78	0x1E0	32	Data lane 0 TSH_SETTLE register
REG7A	0x1E8	32	Data lane 0 calibration reception register
REG98	0x260	32	Data lane 1 TSH_SETTLE register
REG9A	0x268	32	Data lane 1 calibration reception register

REG00

Data lane and clock lane enable register

Bit #	R/W	Name	Reset	Description
2	R/W	LANE_EN0	0x0	Enable D-PHY lane 0: active high
3	R/W	LANE_EN1	0x0	Enable D-PHY lane 1: active high
6	R/W	LANE_CLK_EN	0x0	Enable D-PHY clock lane: active high

REG0D

Digital clock sample manual phase selection register

Bit #	R/W	Name	Reset	Description
2:0	R/W	SAMPLE_PHASE	0x0	Manual phase selection for digital clock sample: 0 to 7, 0 is earliest, 7 is the latest, phase step ~40ps

REG0E

Clock lane manual phase selection register

Bit #	R/W	Name	Reset	Description
6:4	R/W	CLOCK_PHASE	0x3	Manual phase selection for clock lane: 0 to 7, 0 is earliest, 7 is the latest, phase step ~40ps

REG0F

Data lanes manual phase selection register

Bit #	R/W	Name	Reset	Description
2:0	R/W	DATA0_PHASE	0x3	Manual phase selection for data lane 0: 0 to 7, 0 is earliest, 7 is the latest, phase step ~40ps
5:3	R/W	DATA1_PHASE	0x3	Manual phase selection for data lane 1: 0 to 7, 0 is earliest, 7 is the latest, phase step ~40ps

REG12

Reverse digital sample clock register

Bit #	R/W	Name	Reset	Description
7	R/W	SAMPLE_REVERSE	0x0	Set to 1 to reverse the digital sample clock

REG20

Digital reset register

Bit #	R/W	Name	Reset	Description
0	R/W	DIG_RSTN	0x1	Set to 0 to reset digital PHY

REG4A

Clock lane continuous mode register

Bit #	R/W	Name	Reset	Description
5:4	R/W	CLK_CONTINOUS	0x0	Continuous clock mode: b00: disabled, b11: enabled

REG58

Clock lane TSH_SETTLE register

Bit #	R/W	Name	Reset	Description
7:0	R/W	THS_SETTLE	0x1B	Configure the count time of the THS_SETTLE by protocol. After count done, D-PHY will begin to receive the high speed data. See the note below for configuration values.

REG5A

Clock lane calibration reception register

Bit #	R/W	Name	Reset	Description
7	R/W	CALIBRATE	0x0	Calibration reception enable - 1'b0 : disable ; - 1'b1 : enable

REG78

Data lane 0 TSH_SETTLE register

Bit #	R/W	Name	Reset	Description
7:0	R/W	THS_SETTLE	0x1B	Configure the count time of the THS_SETTLE by protocol. After count done, D-PHY will begin to receive the high speed data. See the note below for configuration values.

REG7A

Data lane 0 calibration reception register

Bit #	R/W	Name	Reset	Description
7	R/W	CALIBRATE	0x0	Calibration reception enable - 1'b0 : disable ; - 1'b1 : enable

REG98

Data lane 1 TSH_SETTLE register

Bit #	R/W	Name	Reset	Description
7:0	R/W	THS_SETTLE	0x1B	Configure the count time of the THS_SETTLE by protocol. After count done, D-PHY will begin to receive the high speed data. See the note below for configuration values.

REG9A

Data lane 1 calibration reception register

Bit #	R/W	Name	Reset	Description
7	R/W	CALIBRATE	0x0	Calibration reception enable - 1'b0 : disable ; - 1'b1 : enable

Note:

Configuration of THS_SETTLE depending on the frequency (1/UI):

[80-110MHz] 0x02,
[110-150MHz] 0x03,
[150-300MHz] 0x06,
[300-400MHz] 0x08,
[400-500MHz] 0x0B,
[500-600MHz] 0x0E,
[600-700MHz] 0x10,
[700-800MHz] 0x12,
[800-1000MHz] 0x16,
[1000-1200MHz] 0x1E,
[1200-1400MHz] 0x23,
[1400-1500MHz] 0x2D.

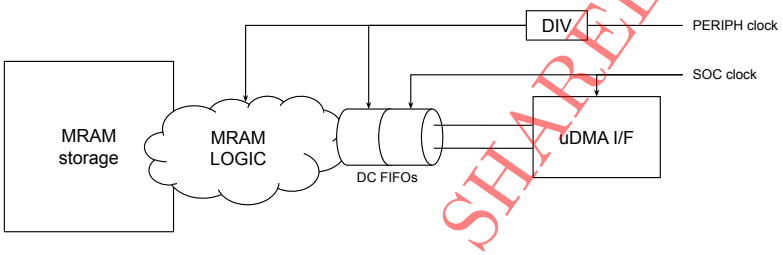
5.3.9 Embedded MRAM

The controller for embedded MRAM supports the following features:

- Configure, chip erase, sector erase, word erase, program and read eMRAM.
- Support for 128 bits = 16 bytes = 4 words access units
- Support for 2S alignement access
- Support for XIP access (when in XIP mode, AUTO mode must be enabled), for read operations only, since program needs erase.
- Support for 2D transfers (when in 2D mode, AUTO mode must be enabled and destination should be set to a UDMA 2D channel).

5.3.9.1 Clocking

The figure below shows the clocking scheme of the embedded MRAM peripheral. The peripheral is clocked by the PERIPH clock generated by the FLL, on which an integer division may be applied. Note that the MRAM configuration interface – not shown on the figure – is clocked by SOC clock.



5.3.9.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_DEST	0x0	32	Stream ID for the uDMA RX channel
TX_DEST	0x4	32	Stream ID for the uDMA TX channel
TRANS_MODE	0x8	32	UDMA transaction mode
TRANS_ADDR	0xC	32	UDMA transaction address
TRANS_SIZE	0x10	32	UDMA transaction size
TRANS_CFG	0x14	32	Start of TX/RX transaction
EXT_ADDR	0x18	32	Target MRAM address
STATUS	0x1C	32	MRAM status
MODE	0x20	32	MRAM mode
ERASE_ADDR	0x24	32	Erase address
ERASE_SIZE	0x28	32	Erase size
CLK_DIV	0x2C	32	Clock divider
ISR	0x30	32	IRQ status register
IER	0x34	32	IRQ enable register
LINE_2D	0x3C	32	MRAM 2D line size
STRIDE_2D	0x40	32	MRAM 2D stride
ENABLE_2D	0x44	32	2D mode enable
TIMING_CFG	0x48	32	Timing configuration

RX_DEST

Stream ID for the uDMA RX channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the uDMA RX channel. Default is 0xFF (disabled)

TX_DEST

Stream ID for the uDMA TX channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST	0xFF	Stream ID for the uDMA TX channel. Default is 0xFF (disabled)

TRANS_MODE

UDMA transaction mode

Bit #	R/W	Name	Reset	Description
0	R/W	AUTO_ENA	0x0	Enables AUTO transfer mode, in which the uDMA transfers are configured automatically, according to register parameters, instead of using SW configuration of uDMA: b0: disabled; b1: enabled
1	R/W	XIP_EN	0x0	Enables XIP transfer mode, in which the uDMA transfers are configured automatically, using XIP parameters, instead of using SW configuration of uDMA: b0: disabled; b1: enabled. Requires AUTO mode
3:2	R	RESERVED	0x0	–
4	R/W	XIP_HALTED	0x0	In XIP mode, halt XIP refill until SW clears this bit: b0: XIP running; b1: XIP halted
5	R/W	XIP_AUTO_HALTED	0x0	Automatically halt XIP refill when an erase operation is ongoing: b0: disabled; b1: enabled

TRANS_ADDR

UDMA transaction address

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDR	0x0	UDMA transfer address when transfer mode is AUTO

TRANS_SIZE

UDMA transaction size

Bit #	R/W	Name	Reset	Description
20:0	R/W	SIZE	0x0	Transfer size when transfer mode is AUTO

TRANS_CFG

Start of TX/RX transaction

Bit #	R/W	Name	Reset	Description
0	R/W	RXTX	0x0	Transfer type: b0: TX; b1: RX
1	R/W	VALID	0x0	Write b0 to clear transfer, write b1 to start transfer. Read always returns 0

EXT_ADDR

Target MRAM address

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDR	0x0	Target address in MRAM (bits 21 to 31 are ignored)

STATUS

MRAM status

Bit #	R/W	Name	Reset	Description
0	R	ERASE_BUSY	0x0	Erase busy: reads b1 when an MRAM erase operation is ongoing
1	R	TX_BUSY	0x0	TX busy: reads b1 when a TX transfer is ongoing
2	R	RX_BUSY	0x0	RX busy: reads b1 when an RX transfer is ongoing
3	R	RESERVED	0x0	–
4	R	UE_ERR	0x0	Unrecoverable error: reads b1 if any unrecoverable error is detected. Not 100% accurate for 3+ bit errors (84% detection rate for 3-bit errors)
5	R	EC_ERR	0x0	ECC-corrected errors: reads b0 if no error is detected, reads b1 if a 1- or 2-bit error has been corrected

MODE

MRAM mode

Bit #	R/W	Name	Reset	Description
0	R/W	ECCBYPS	0x0	Controls MRAM ECC bypass signal
1	R/W	DPD	0x0	Controls MRAM deep power down signal
3	R/W	TMEN	0x0	Controls MRAM TMEN signal
4	R/W	NVR	0x0	Controls MRAM NVR signal
5	R/W	RSTb	0x0	Controls MRAM RSTb signal
6	R/W	RETb	0x0	Controls MRAM RETb signal
7	R/W	PORb	0x0	Controls MRAM PORb signal
15:8	R/W	OPERATION	0x0	Configure MRAM operation: h00: power up; h01: configure trimming; h02: program; h04: chip erase; h08: sector erase; h10 word erase; h20: power down; h40: read

ERASE_ADDR

Erase address

Bit #	R/W	Name	Reset	Description
20:0	R/W	ADDR	0x0	Erase address (bits 0 to 3 are ignored)

ERASE_SIZE

Erase size

Bit #	R/W	Name	Reset	Description
6:0	R/W	SIZE	0x0	Erase Size

CLK_DIV

Clock divider

Bit #	R/W	Name	Reset	Description
7:0	R/W	DATA	0x0	Clock divider ratio (0-255): PERIPH clock frequency is divided by DATA
8	R/W	VALID	0x0	Set to b1 to have divider ratio taken into account; automatically reset to b0 by the system once the ratio is taken into account
9	R/W	ENABLE	0x0	Enable MRAM clock: b0: clock gated; b1: clock enabled

ISR

IRQ status register

Bit #	R/W	Name	Reset	Description
0	R	ERASE_DONE	0x0	End-of-erase IRQ status
1	R	PROGRAM_DONE	0x0	End-of-program IRQ status
2	R	TRIM_CONFIG_-DONE	0x0	End-of-trimming-configuration IRQ status
3	R	RX_DONE	0x0	End-of-read IRQ status

IER

IRQ enable register

Bit #	R/W	Name	Reset	Description
0	R/W	ERASE_EN	0x0	Enable end-of-erase IRQ
1	R/W	PROGRAM_EN	0x0	Enable end-of-program IRQ
2	R/W	TRIM_CONFIG_EN	0x0	Enable end-of-trimming-configuration IRQ
3	R/W	RX_DONE_EN	0x0	Enable end-of-read IRQ
4	R/W	XIP_ERASE_EN	0x0	Enable end-of-XIP erase IRQ
5	R/W	XIP_PROGRAM_EN	0x0	Enable end-of-XIP program IRQ
6	R/W	XIP_TRIM_CONFIG_EN	0x0	Enable end-of-XIP trimming-configuration IRQ
7	R/W	RX_XIP_DONE_EN	0x0	Enable end-of-XIP read IRQ

LINE_2D

MRAM 2D line size

Bit #	R/W	Name	Reset	Description
31:0	R/W	LINE	0x0	Length of MRAM 2D line when in 2D mode. For example, ADDR = START_ADDR + i * BURST_STRIDE. Normally, LINE >= BURST_SIZE

STRIDE_2D

MRAM 2D stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Length of MRAM 2D stride when in 2D mode. For example, ADDR = START_ADDR + i * BURST_STRIDE. Normally, STRIDE >= BURST_SIZE

ENABLE_2D

2D mode enable

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	MRAM 2D mode enable: b0: 2D mode disabled; b1: 2D mode enabled

TIMING_CFG

Timing configuration

Bit #	R/W	Name	Reset	Description
2:0	R/W	STROBE_TIME_CNT	0x2	Strobe timing counter
5:3	R/W	GO_SUP_TIME_CNT	0x4	Power supply timing counter
8:6	R/W	MEN_TIME_CNT	0x4	MRAM EN timing counter
15:9	R/W	RW_TIME_CNT	0x78	Latency from write to read timing counter
18:16	R/W	ADS_TIME_CNT	0x4	Address setup timing counter
28:19	R/W	PGS_TIME_CNT	0x320	Program setup timing counter
31:29	R/W	PROG_TIME_CNT	0x8	Program minimum pulse width timing counter

5.3.10 Timestamping

UDMA timestamping module associates a timestamp (built upon a 24-bit counter) to a specific event coming from either SOC events 0 to 3, GPIOs 0 to 63, or AUX events 0 to 11. AUX are associated to events coming from SFU or from SAI WS signals. The maximum allowed frequency for GPIO and PWM is 1Mhz.

The timestamp module outputs data each time one of the following conditions is met:

- One of the 4 SOC events is active:
  Generated output is b0000_00xx_cccc_cccc_cccc_cccc_cccc_cccc, where xx is the index of the arbritrated event, and cccc_…_cccc is the counter value. The channel ID used for the transaction is the one configured for this event in the REG_EVENT register.
- An output channel fulfil the conditions configured in its REG_SETUP_CH* register, i.e.
  - either it is configured to trigger on one of the 11 AUX events, and this event is active,
  - or it is configured to trigger on one of the 64 GPIOs, which has toggled according to selected edge(s).

  Output data is b1000_0xxx_cccc_cccc_cccc_cccc_cccc_cccc, where xxx is the index of the channel, and cccc_…_cccc is the counter value. The channel ID used for the transaction is the one configured in the REG_SETUP_CH* register for this channel.

In case of simultaneous events, a mechanism ensures that an output is produced for each of them, so that no event is lost.

Activation of the counting can be controlled in two ways:

- Using the REG_CMD register to start, restart (which resets counter value to 0) or stop the counter.
- Using a GPIO as a trigger to start/restart the counter, as configured in REG_SETUP_CNT register.

Once started, the counter increments its value, based on the selected source clock, on which the PRESCALER dividing factor is applied (see REG_CLK_CFG register). When the counter reaches its maximum value, it wraps to 0 and continues counting.

5.3.10.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
REG_CMD	0x0	32	Counter start/stop/reset
REG_SETUP_CNT	0x4	32	Configuration of counter start through GPIO
REG_SETUP_CH0	0x8	32	Channel 0 configuration (AUX or GPIOs events)
REG_SETUP_CH1	0xC	32	Channel 1 configuration (AUX or GPIOs events)
REG_SETUP_CH2	0x10	32	Channel 2 configuration (AUX or GPIOs events)
REG_SETUP_CH3	0x14	32	Channel 3 configuration (AUX or GPIOs events)
REG_SETUP_CH4	0x18	32	Channel 4 configuration (AUX or GPIOs events)
REG_SETUP_CH5	0x1C	32	Channel 5 configuration (AUX or GPIOs events)
REG_SETUP_CH6	0x20	32	Channel 6 configuration (AUX or GPIOs events)
REG_SETUP_CH7	0x24	32	Channel 7 configuration (AUX or GPIOs events)
REG_EVENT	0x28	32	SOC events timestamp configuration
REG_CLK_CFG	0x2C	32	Counting clock configuration

REG_CMD

Counter start/stop/reset

Bit #	R/W	Name	Reset	Description
0	W	CNT_CLR	0x0	Write 1 to reset the counter to 0 and start/restart it
1	W	CNT_STOP	0x0	Write 1 to reset the counter to 0 and stop it

REG_SETUP_CNT

Configuration of counter start through GPIO

Bit #	R/W	Name	Reset	Description
5:0	R/W	EXT_SEL	0x00	ID of the GPIO used to start/restart the counter
7:6	R/W	EXT_TYPE	0x0	GPIO condition to trigger counter start/restart: 0: rising edge; 1: falling edge; 2: both edges; 3: never
8	R/W	EXT_EN	0x0	If 1, GPIO-controlled start/restart of the counter is activated

REG_SETUP_CH0

Channel 0 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 0, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 0: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 0
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 0

REG_SETUP_CH1

Channel 1 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 1, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 1: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 1
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 1

REG_SETUP_CH2

Channel 2 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 2, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 2: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 2
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 2

REG_SETUP_CH3

Channel 3 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 3, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 3: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 3
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 3

REG_SETUP_CH4

Channel 4 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 4, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 4: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 4
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 4

REG_SETUP_CH5

Channel 5 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 5, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 5: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 5
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 5

REG_SETUP_CH6

Channel 6 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 6, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 6: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 6
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 6

REG_SETUP_CH7

Channel 7 configuration (AUX or GPIOs events)

Bit #	R/W	Name	Reset	Description
5:0	R/W	INPUT_SEL	0x00	Select event input index for channel 7, depending on INPUT_TYPE: 0 to 63 for GPIOs, 0 to 7 for SFU events, 8 to 10 for WS signal of SAI0 to SAI2
7:6	R/W	INPUT_TYPE	0x0	Event source for channel 7: 0: GPIO rising edge; 1: GPIO falling edge; 2: GPIO both edges; 3: AUX event (SFU or SAI WS)
8	R/W	INPUT_EN	0x0	Enable timestamping for channel 7
23:16	R/W	DEST_ID	0x00	UDMA destination ID for channel 7

REG_EVENT

SOC events timestamp configuration

Bit #	R/W	Name	Reset	Description
7:0	R/W	DEST_ID_EVT_0	0xFF	UDMA destination ID for SOC event 0
15:8	R/W	DEST_ID_EVT_1	0xFF	UDMA destination ID for SOC event 1
23:16	R/W	DEST_ID_EVT_2	0xFF	UDMA destination ID for SOC event 2
31:24	R/W	DEST_ID_EVT_3	0xFF	UDMA destination ID for SOC event 3

REG_CLK_CFG

Counting clock configuration

Bit #	R/W	Name	Reset	Description
1:0	R/W	CLK_MUX	0x0	Select clock source: b00: PWM; b01: GPIO; b10: REF FAST clock
2	R/W	CLK_MUX_EN	0x0	Enable the clock mux: if b0, internal SOC clock is used
6:4	R/W	PWM_SEL	0x0	Select PWM clock source among the 8 possible PWM
13:8	R/W	GPIO_SEL	0x0	Select GPIO clock source among the 64 possible GPIOs
23:16	R/W	PRESCALER	0x0	Clock counter prescaler

5.3.11 Dual-Core AES

Dual-core AES supports on-the-fly AES encryption/decryption of 2 interleaved data streams using 2 different encryption keys. It is usefull for memory controllers and it supports XIP.

5.3.11.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
KEY0_0	0x0	32	Word 0 of encryption key for core 0
KEY0_1	0x4	32	Word 1 of encryption key for core 0
KEY0_2	0x8	32	Word 2 of encryption key for core 0
KEY0_3	0xC	32	Word 3 of encryption key for core 0
KEY0_4	0x10	32	Word 4 of encryption key for core 0
KEY0_5	0x14	32	Word 5 of encryption key for core 0
KEY0_6	0x18	32	Word 6 of encryption key for core 0
KEY0_7	0x1C	32	Word 7 of encryption key for core 0
IV0_0	0x20	32	Word 0 of encrypted block initial value for core 0
IV0_1	0x24	32	Word 1 of encrypted block initial value for core 0
IV0_2	0x28	32	Word 2 of encrypted block initial value for core 0
IV0_3	0x2C	32	Word 3 of encrypted block initial value for core 0
KEY1_0	0x30	32	Word 0 of encryption key for core 1
KEY1_1	0x34	32	Word 1 of encryption key for core 1
KEY1_2	0x38	32	Word 2 of encryption key for core 1
KEY1_3	0x3C	32	Word 3 of encryption key for core 1
KEY1_4	0x40	32	Word 4 of encryption key for core 1
KEY1_5	0x44	32	Word 5 of encryption key for core 1
KEY1_6	0x48	32	Word 6 of encryption key for core 1
KEY1_7	0x4C	32	Word 7 of encryption key for core 1
IV1_0	0x50	32	Word 0 of encrypted block initial value for core 1
IV1_1	0x54	32	Word 1 of encrypted block initial value for core 1
IV1_2	0x58	32	Word 2 of encrypted block initial value for core 1
IV1_3	0x5C	32	Word 3 of encrypted block initial value for core 1
DEST	0x60	32	RX and TX destination channels
SETUP0	0x64	32	Core 0 setup
SETUP1	0x68	32	Core 1 setup
CFG	0x6C	32	AES data flow configuration

KEY0_0

Word 0 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_1

Word 1 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_2

Word 2 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_3

Word 3 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_4

Word 4 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_5

Word 5 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_6

Word 6 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_7

Word 7 of encryption key for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

IV0_0

Word 0 of encrypted block initial value for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_1

Word 1 of encrypted block initial value for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_2

Word 2 of encrypted block initial value for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_3

Word 3 of encrypted block initial value for core 0

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

KEY1_0

Word 0 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_1

Word 1 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_2

Word 2 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_3

Word 3 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_4

Word 4 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_5

Word 5 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_6

Word 6 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY1_7

Word 7 of encryption key for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

IV1_0

Word 0 of encrypted block initial value for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV1_1

Word 1 of encrypted block initial value for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV1_2

Word 2 of encrypted block initial value for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV1_3

Word 3 of encrypted block initial value for core 1

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

DEST

RX and TX destination channels

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_DEST	0xFF	Stream ID for the RX uDMA channel. Default is 0xFF (channel disabled)
15:8	R/W	TX_DEST	0xFF	Stream ID for the TX uDMA channel. Default is 0xFF (channel disabled)

SETUP0

Core 0 setup

Bit #	R/W	Name	Reset	Description
0	R	KEY_INIT	0x0	Is set to 1 when the key configuration is finished
1	R/W	KEY_TYPE	0x0	Key type: b0: 128 bits; b1: 256 bits
2	R/W	ENC_DEC	0x0	Operation type: b0: decryption; b1: encryption
3	R/W	ECB_CBC	0x0	Encryption type: b0: ECB; b1: CBC
4	W	BLOCK_RST	0x0	Write b1 to reset AES core 0
5	R/W	QK_KEY_EN	0x0	Use quiddikey key generation
8	W	FIFO_CLR	0x0	Write b1 to clear data FIFO

SETUP1

Core 1 setup

Bit #	R/W	Name	Reset	Description
0	R	KEY_INIT	0x0	Is set to 1 when the key configuration is finished
1	R/W	KEY_TYPE	0x0	Key type: b0: 128 bits; b1: 256 bits
2	R/W	ENC_DEC	0x0	Operation type: b0: decryption; b1: encryption
3	R/W	ECB_CBC	0x0	Encryption type: b0: ECB; b1: CBC
4	W	BLOCK_RST	0x0	Write b1 to reset AES core 1
5	R/W	QK_KEY_EN	0x0	Use quiddikey key generation
8	W	FIFO_CLR	0x0	Write b1 to clear data FIFO

CFG

AES data flow configuration

Bit #	R/W	Name	Reset	Description
1:0	R/W	MODE	0x0	Transfer mode for core 0: b00: memory to memory; b01: stream to memory; b10: memory to stream; b11: stream to stream — reverse configuration is used for core 1. When used with memory controller, must be set to b10 (core 0: memory to stream for encryption; core 1: stream to memory for decryption)

5.3.12 AES

AES unit provides on-the-fly AES encryption/decryption of a single flow of data.

5.3.12.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
KEY0_0	0x0	32	Word 0 of encryption key
KEY0_1	0x4	32	Word 1 of encryption key
KEY0_2	0x8	32	Word 2 of encryption key
KEY0_3	0xC	32	Word 3 of encryption key
KEY0_4	0x10	32	Word 4 of encryption key
KEY0_5	0x14	32	Word 5 of encryption key
KEY0_6	0x18	32	Word 6 of encryption key
KEY0_7	0x1C	32	Word 7 of encryption key
IV0_0	0x20	32	Word 0 of encrypted block initial value
IV0_1	0x24	32	Word 1 of encrypted block initial value
IV0_2	0x28	32	Word 2 of encrypted block initial value
IV0_3	0x2C	32	Word 3 of encrypted block initial value
DEST	0x30	32	RX and TX destination channels
SETUP	0x34	32	Core setup
CFG	0x38	32	AES data flow configuration

KEY0_0

Word 0 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_1

Word 1 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_2

Word 2 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_3

Word 3 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_4

Word 4 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_5

Word 5 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_6

Word 6 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

KEY0_7

Word 7 of encryption key

Bit #	R/W	Name	Reset	Description
31:0	W	KEY_WORD	0x00000000	Value of the corresponding word of encryption key

IV0_0

Word 0 of encrypted block initial value

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_1

Word 1 of encrypted block initial value

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_2

Word 2 of encrypted block initial value

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

IV0_3

Word 3 of encrypted block initial value

Bit #	R/W	Name	Reset	Description
31:0	W	BLOCK_WORD	0x00000000	Value of the corresponding word of the initial encrypted block

DEST

RX and TX destination channels

Bit #	R/W	Name	Reset	Description
7:0	R/W	RX_DEST	0xFF	Stream ID for the RX uDMA channel. Default is 0xFF (channel disabled)
15:8	R/W	TX_DEST	0xFF	Stream ID for the TX uDMA channel. Default is 0xFF (channel disabled)

SETUP

Core setup

Bit #	R/W	Name	Reset	Description
0	R	KEY_INIT	0x0	Is set to 1 when the key configuration is finished
1	R/W	KEY_TYPE	0x0	Key type: b0: 128 bits; b1: 256 bits
2	R/W	ENC_DEC	0x0	Operation type: b0: decryption; b1: encryption
3	R/W	ECB_CBC	0x0	Encryption type: b0: ECB; b1: CBC
4	W	BLOCK_RST	0x0	Write b1 to reset AES core
5	R/W	QK_KEY_EN	0x0	Use quiddikey key generation
8	W	FIFO_CLR	0x0	Write b1 to clear data FIFO

CFG

AES data flow configuration

Bit #	R/W	Name	Reset	Description
1:0	R/W	MODE	0x0	Transfer mode: b00: memory to memory; b01: stream to memory; b10: memory to stream; b11: stream to stream

5.3.13 Smart Filtering Unit (SFU)

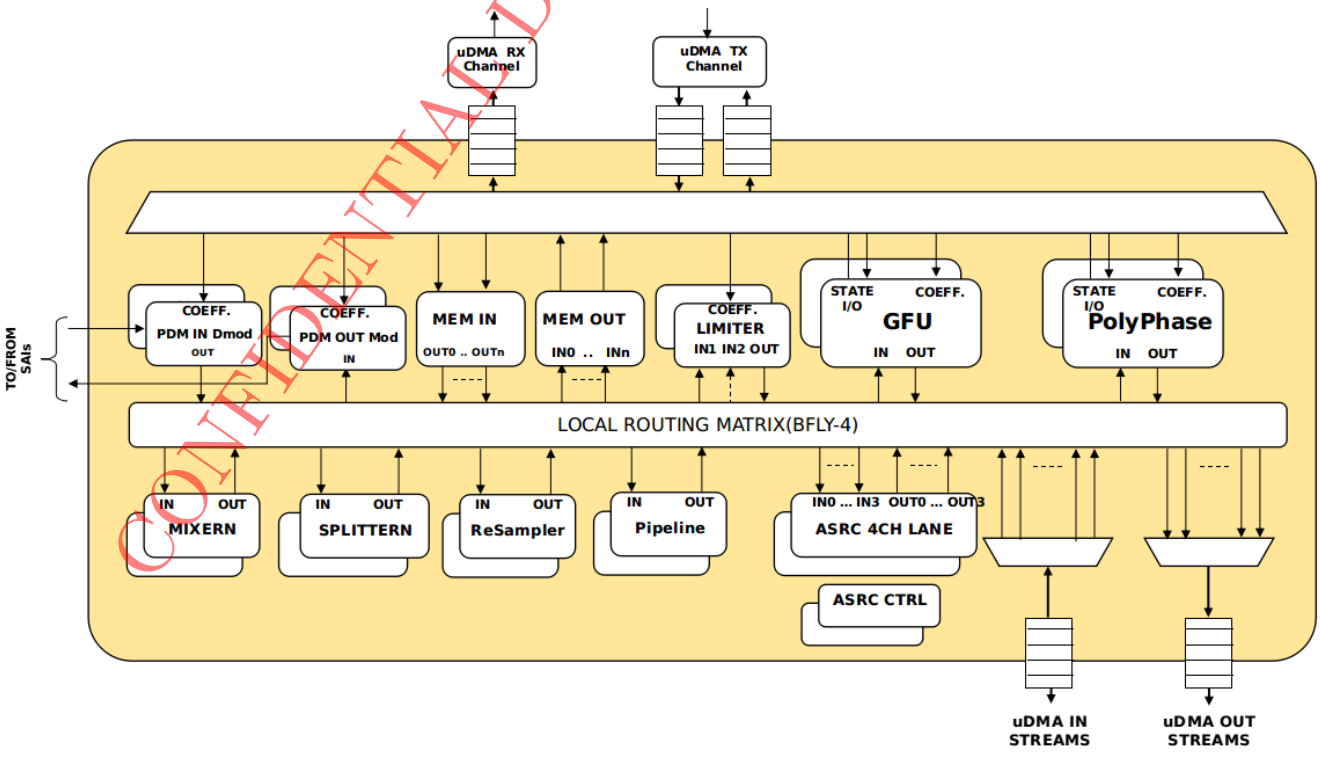
The Smart Filtering Unit aims at significantly enhancing GAP9 capability to support audio graph transformations with the following motivations:

- Offload CPUs from heavy duty recurrent tasks such as asynchronous sample rate conversion.
- Lowering power consumption thanks to specialization.
- Enable ultra low latency processing with response time below sampling rate.

The SFU is strongly linked to the serial audio interfaces and to the uDMA core controller. It can be qualified as a configurable stream engine, and as such it is fully decoupled from the rest of Gap9 SoC domain, it operates in its own frequency domain with inbound and outbound traffic passing through bisynchronous FIFOs. This property allows the SFU to possibly operate at a much higher frequency than the rest of the chip when needed, avoiding high toggling rate in the rest of the chip.

Inside the SFU there are several classes of sub blocks with different roles (see next figure):

- Configurable transformer blocks to perform the actual filtering.
- Sources/sinks blocks for data coming from/sent to outside of the SFU (memory, peripherals or SAI-PDM direct connection).
- Splitter/joiner blocks.
- Traffic shaper blocks (envelope, volume, ...).
- A routing structure to move samples to/from SFU blocks input/output ports.
- A sequencer block to control internal data flow inside configurable blocks and to orchestrate state changes/reconfiguration in an orderly manner.



The current implementation contains the following SFU blocks:

- 4× GFU: Global Filtering Unit, which supports 8 configurable filtering patterns, and multiple contexts.
- 12× Demod: PDM to PCM demodulator.
- 6× Mod: PCM to PDM modulator.
- 6× Limiter: signal limiter, with mute/unmute and volume control.
- 4× Splitter: 1 input to N outputs, supporting multiple contexts.
- 4× Adder/Mixer: N inputs to 1 output mixer with controllable mixing, supporting multiple contexts.
- 4× PolyPhase: cascaded polyphase filter.
- 12× ASRC: synchronous sample rate converter, embedding 3 Fin/Fout frequency trackers.
- 8× MemIn: L2 memory to SFU block input.
- 8× MemOut: SFU block output to L2 memory.
- 8× StreamIn: Peripheral to SFU block input, through uDMA.
- 8× StreamOut: SFU block output to Peripheral through uDMA.

More details on SFU operation and programming are given in a separate document. Contact GreenWaves Technologies for more information.

5.3.13.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
GRAPH_PTR	0x0	32	Pointer to graph configuration
GRAPH_CMD	0x4	32	Graph command register
CLOCK_PTR	0x8	32	Pointer to clock configuration
CLOCK_CMD	0xC	32	Clock command register
SFU_STATUS	0x10	32	Status of graph and clocks commands
MEM_IN_STATUS	0x14	32	Status of memory IN interfaces
LIMITER_MUTE	0x18	32	Limiter mute/unmute control
VOLUME_INDEX	0x1C	32	Control of mute/unmute
VOLUME_VALUE	0x20	32	Control of linear volume
CLK_MONITOR_0	0x24	32	Control of clock monitors 0 to 3
CLK_MONITOR_1	0x28	32	Control of clock monitors 4 to 7
OUT_MUTE	0x2C	32	Control of output channel mute
AUDIO_CLK_CFG_0	0x30	32	Control audio clock generator 0
AUDIO_CLK_CFG_1	0x34	32	Control audio clock generator 1
AUDIO_CLK_CFG_2	0x38	32	Control audio clock generator 2
AUDIO_CLK_CFG_3	0x3C	32	Control audio clock generator 3
ASRC_RATIO_0	0x40	32	ASRC0 conversion ratio
ASRC_RATIO_1	0x44	32	ASRC1 conversion ratio
ASRC_RATIO_2	0x48	32	ASRC2 conversion ratio
MEM_IN_0_CNT	0x58	32	Memory input counter 0
MEM_IN_1_CNT	0x5C	32	Memory input counter 1
MEM_IN_2_CNT	0x60	32	Memory input counter 2
MEM_IN_3_CNT	0x64	32	Memory input counter 3
MEM_IN_4_CNT	0x68	32	Memory input counter 4
MEM_IN_5_CNT	0x6C	32	Memory input counter 5
MEM_IN_6_CNT	0x70	32	Memory input counter 6
MEM_IN_7_CNT	0x74	32	Memory input counter 7

GRAPH_PTR

Pointer to graph configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Address of graph configuration in memory

GRAPH_CMD

Graph command register

Bit #	R/W	Name	Reset	Description
0	W	LOAD	0x0	Write b1 to start graph load
1	W	RECONF	0x0	Write b1 to start graph reconfiguration
2	W	UNLOAD	0x0	Write b1 to start graph unload
3	W	SAVE	0x0	Write b1 to start graph save
4	W	SET_CURRENT	0x0	Sets the current graph (used for status read)

CLOCK_PTR

Pointer to clock configuration

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Address of clock configuration in memory

CLOCK_CMD

Clock command register

Bit #	R/W	Name	Reset	Description
0	W	LOAD	0x0	Write b1 to start clock load
1	W	UNLOAD	0x0	Write b1 to start clock unload

SFU_STATUS

Status of graph and clocks commands

Bit #	R/W	Name	Reset	Description
0	R	CLOCK_LOAD	0x0	Bit is set to 1 when clock load is ongoing
1	R	GRAPH_LOAD	0x0	Bit is set to 1 when graph load is ongoing
2	R	GRAPH_UNLOAD	0x0	Bit is set to 1 when graph unload is ongoing
3	R	GRAPH_RECONF	0x0	Bit is set to 1 when graph reconfiguration is ongoing
4	R	GRAPH_SAVE	0x0	Bit is set to 1 when graph save is ongoing
5	R	GRAPH_SET_CUR	0x0	Bit is set to 1 when current graph is being set
8:6	R	ASRC_LOCK	0x0	Lock status of the 3 ASRCs: bit i is set to 1 when frequency tracking of ASRCi is locked
9	R	GRAPH_BUSY	0x0	Bit is set to 1 when current graph is busy

MEM_IN_STATUS

Status of memory IN interfaces

Bit #	R/W	Name	Reset	Description
7:0	R/W	STATUS	0x0	When reading, bit i give the status of MemIn interface i: b0: interface OK; b1: buffer has ended. Writing b1 to bit i restarts the MemIn interface i (e.g. after buffer restart)

LIMITER_MUTE

Limiter mute/unmute control

Bit #	R/W	Name	Reset	Description
0	R/W	MUTE_LIM_0	0x0	Enable mute 0: b0: mute disabled; b1: mute enabled
1	R/W	MUTE_LIM_1	0x0	Enable mute 1: b0: mute disabled; b1: mute enabled
2	R/W	MUTE_LIM_2	0x0	Enable mute 2: b0: mute disabled; b1: mute enabled
3	R/W	MUTE_LIM_3	0x0	Enable mute 3: b0: mute disabled; b1: mute enabled
4	R/W	MUTE_LIM_4	0x0	Enable mute 4: b0: mute disabled; b1: mute enabled
5	R/W	MUTE_LIM_5	0x0	Enable mute 5: b0: mute disabled; b1: mute enabled

VOLUME_INDEX

Control of mute/unmute

Bit #	R/W	Name	Reset	Description
4:0	R/W	INDEX	0x0	Index of volume setting accessed by VOLUME_VALUE register

VOLUME_VALUE

Control of linear volume

Bit #	R/W	Name	Reset	Description
25:0	R/W	VOLUME	0x0	Value of volume (linear)
31:26	R/W	SCALING_V	0x0	Value in bits for the scaling (bit 5 is the direction)

CLK_MONITOR_0

Control of clock monitors 0 to 3

Bit #	R/W	Name	Reset	Description
4:0	R/W	SEL0	0x0	Monitored clock selector (see Clock select table below)
7	R/W	EN0	0x0	Set to b1 to enable monitoring
12:8	R/W	SEL1	0x0	Monitored clock selector (see Clock select table below
15	R/W	EN1	0x0	Set to b1 to enable monitoring
20:16	R/W	SEL2	0x0	Monitored clock selector (see Clock select table below
23	R/W	EN2	0x0	Set to b1 to enable monitoring
28:24	R/W	SEL3	0x0	Monitored clock selector (see Clock select table below
31	R/W	EN3	0x0	Set to b1 to enable monitoring

CLK_MONITOR_1

Control of clock monitors 4 to 7

Bit #	R/W	Name	Reset	Description
4:0	R/W	SEL0	0x0	Monitored clock selector (see Clock select table below)
7	R/W	EN0	0x0	Set to b1 to enable monitoring
12:8	R/W	SEL1	0x0	Monitored clock selector (see Clock select table below
15	R/W	EN1	0x0	Set to b1 to enable monitoring
20:16	R/W	SEL2	0x0	Monitored clock selector (see Clock select table below
23	R/W	EN2	0x0	Set to b1 to enable monitoring
28:24	R/W	SEL3	0x0	Monitored clock selector (see Clock select table below
31	R/W	EN3	0x0	Set to b1 to enable monitoring

OUT_MUTE

Control of output channel mute

Bit #	R/W	Name	Reset	Description
7:0	R/W	MEM_OUT	0x0	Mutes corresponding MemOut channel
15:8	R/W	STREAM_OUT	0x0	Mutes corresponding StreamOut channel
18:16	R/W	PDM_OUT	0x0	Mutes corresponding PDMOut channel

AUDIO_CLK_CFG_0

Control audio clock generator 0

Bit #	R/W	Name	Reset	Description
15:0	R/W	DIV	0x0	Division factor for audio clock
16	R/W	EN	0x0	Enable: b0: audio clock disabled; b1: audio clock enabled

AUDIO_CLK_CFG_1

Control audio clock generator 1

Bit #	R/W	Name	Reset	Description
15:0	R/W	DIV	0x0	Division factor for audio clock
16	R/W	EN	0x0	Enable: b0: audio clock disabled; b1: audio clock enabled

AUDIO_CLK_CFG_2

Control audio clock generator 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	DIV	0x0	Division factor for audio clock
16	R/W	EN	0x0	Enable: b0: audio clock disabled; b1: audio clock enabled

AUDIO_CLK_CFG_3

Control audio clock generator 3

Bit #	R/W	Name	Reset	Description
15:0	R/W	DIV	0x0	Division factor for audio clock
16	R/W	EN	0x0	Enable: b0: audio clock disabled; b1: audio clock enabled

ASRC_RATIO_0

ASRC0 conversion ratio

Bit #	R/W	Name	Reset	Description
25:0	R/W	RATIO	0x0	Conversion ratio

ASRC_RATIO_1

ASRC1 conversion ratio

Bit #	R/W	Name	Reset	Description
25:0	R/W	RATIO	0x0	Conversion ratio

ASRC_RATIO_2

ASRC2 conversion ratio

Bit #	R/W	Name	Reset	Description
25:0	R/W	RATIO	0x0	Conversion ratio

MEM_IN_0_CNT

Memory input counter 0

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_1_CNT

Memory input counter 1

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_2_CNT

Memory input counter 2

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_3_CNT

Memory input counter 3

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_4_CNT

Memory input counter 4

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_5_CNT

Memory input counter 5

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_6_CNT

Memory input counter 6

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

MEM_IN_7_CNT

Memory input counter 7

Bit #	R/W	Name	Reset	Description
20:0	R	CNT	0x0	Reports how many samples have been pushed to the SFU from this MemIn interface

5.3.13.2 Clock Select Table

Clock ID	Clock Name	Description
0	SAI0	Sample clock of SAI0
1	SAI1	Sample clock of SAI1
2	SAI2	Sample clock of SAI2
3	PDM0	Sample clock of PDM demodulator 0
4	PDM1	Sample clock of PDM demodulator 1
5	PDM2	Sample clock of PDM demodulator 2
6	PDM3	Sample clock of PDM demodulator 3
7	PDM4	Sample clock of PDM demodulator 4
8	PDM5	Sample clock of PDM demodulator 5
9	PDM6	Sample clock of PDM demodulator 6
10	PDM7	Sample clock of PDM demodulator 7
11	PDM8	Sample clock of PDM demodulator 8
12	PDM9	Sample clock of PDM demodulator 9
13	PDM10	Sample clock of PDM demodulator 10
14	PDM11	Sample clock of PDM demodulator 11
15	PWM0	derived from SOC clk
16	PWM1	derived from SOC clk
17	PWM2	derived from SOC clk
18	PWM3	derived from SOC clk
19	AUD0	derived from REF FAST clk (divided by power of 2)
20	AUD1	derived from REF FAST clk (divided by power of 2)
21	AUD2	derived from REF FAST clk (divided by power of 2)
22	AUD3	derived from REF FAST clk (divided by power of 2)
23	VIRT	define the clock as virtual

5.3.14 Fixed-/Floating-Point Converter

The FFC unit handles on-the-fly conversion of fixed-point values to floating-point values and vice-versa.

5.3.14.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
RX_DEST	0x0	32	Stream ID for the uDMA RX channel
TX_DEST	0x4	32	Stream ID for the uDMA TX channel
FL_FORMAT	0x8	32	Floating point format
FP_FORMAT	0xC	32	Fixed point format
FP_PREC	0x10	32	Fixed point precision
FP_SCALE	0x14	32	Fixed point scale
MODE	0x18	32	Direction and I/O mode configuration
TRANS_MODE	0x1C	32	Use auto transfer or not
CONV_NUM	0x20	32	Number of conversions to be done
CONV_TX_ADDR	0x24	32	TX channel address in L2 when auto mode is set
CONV_RX_ADDR	0x28	32	RX channel address in L2 when auto mode is set
IRQ_EN	0x2C	32	Enable or disable transfer interrupt
STATUS	0x30	32	Show if the transfer is finished or not
START	0x34	32	Start processing

RX_DEST

Stream ID for the uDMA RX channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	ID_RX	0xFF	UDMA stream ID for RX channel

TX_DEST

Stream ID for the uDMA TX channel

Bit #	R/W	Name	Reset	Description
7:0	R/W	ID_TX	0xFF	UDMA stream ID for TX channel

FL_FORMAT

Floating point format

Bit #	R/W	Name	Reset	Description
1:0	R/W	FORMAT	0x0	Floating point format: b00: FP16; b01: BFP16; b10-b11: FP32

FP_FORMAT

Fixed point format

Bit #	R/W	Name	Reset	Description
1:0	R/W	FORMAT	0x0	Fixed point format: b00: 8 bits; b01: 16 bits; b10: 24 bits; b11: 32bits

FP_PREC

Fixed point precision

Bit #	R/W	Name	Reset	Description
4:0	R/W	PRECISION	0x0	Precision of fixed point format (cannot exceed fixed point size)

FP_SCALE

Fixed point scale

Bit #	R/W	Name	Reset	Description
15:0	R/W	SCALE	0x0	Fixed point scale (signed number): data are multiplied by 2^{scale} to compute their real value

MODE

Direction and I/O mode configuration

Bit #	R/W	Name	Reset	Description
0	R/W	DIRECTION	0x0	0: Floating point to fixed point; 1: Fixed point to floating point
2:1	R/W	IO_MODE	0x0	b00: MIMO; b01: SIMO; b10: MISO; b11: SISO

TRANS_MODE

Use auto transfer or not

Bit #	R/W	Name	Reset	Description
0	R/W	AUTO_EN	0x0	Set to 1 to enable auto transfer

CONV_NUM

Number of conversions to be done

Bit #	R/W	Name	Reset	Description
31:0	R/W	NUM	0x0	Number of conversions to be done

CONV_TX_ADDR

TX channel address in L2 when auto mode is set

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	In auto mode, gives the address in L2 for TX channel

CONV_RX_ADDR

RX channel address in L2 when auto mode is set

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	In auto mode, gives the address in L2 for RX channel

IRQ_EN

Enable or disable transfer interrupt

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	Set to 1 to enable transfer interrupt

STATUS

Show if the transfer is finished or not

Bit #	R/W	Name	Reset	Description
0	R	DONE	0x0	Is set to 1 when the configured number of conversions have been completed
1	R	BUSY	0x0	Is set to 1 while conversions are ongoing

START

Start processing

Bit #	R/W	Name	Reset	Description
0	W	START	0x0	Write any value to start the processing of conversions according to configuration

5.3.15 Linear Channels

UDMA linear address generators

5.3.15.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_SA_BUF0	0x0	32	Start address of 1st buffer used in legacy mode and continuous mode (HW double buffer)
CFG_SA_BUF1	0x4	32	Start address of 2nd buffer used in continuous mode
CFG_SIZE	0x8	32	Sets the size of the buffers
CFG_CURR_ADDR	0x14	32	Current address
CFG_BYTES_LEFT	0x18	32	Number of bytes left
CFG_CTRL	0x1C	32	Linear generator control

CFG_SA_BUF0

Start address of 1st buffer used in legacy mode and continuous mode (HW double buffer)

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Start address of the 1st buffer (the 8 upper bits are forced to 0x1C)

CFG_SA_BUF1

Start address of 2nd buffer used in continuous mode

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Start address of the 2nd buffer, when in continuous mode (the 8 upper bits are forced to 0x1C)

CFG_SIZE

Sets the size of the buffers

Bit #	R/W	Name	Reset	Description
19:0	R/W	SIZE	0x0	Buffer size (i.e. transfer size) in bytes

CFG_CURR_ADDR

Current address

Bit #	R/W	Name	Reset	Description
31:0	R	ADDRESS	0x0	Value of current address in the buffer (the 8 upper bits are always 0x1C)

CFG_BYTES_LEFT

Number of bytes left

Bit #	R/W	Name	Reset	Description
20:0	R	NUM_BYTES	0x0	Number of remaining bytes to transfer

CFG_CTRL

Linear generator control

Bit #	R/W	Name	Reset	Description
0	R/W	CONT	0x0	Enable hardware double buffer support (continuous mode): b0: legacy mode; b1: continuous mode
1	R/W	EN	0x0	Write b1 to start a transfer or enqueue a new transfer if one is already running. On read: b0: no ongoing transfer; b1: ongoing transfer
4	W	STOP	0x0	Write b1 to stop the current transfer and reset the address generator

5.3.16 2D Channels

UDMA 2D address generators

5.3.16.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_SA_BUF0	0x0	32	Start address of 1st buffer used in legacy mode and continuous mode (HW double buffer)
CFG_SA_BUF1	0x4	32	Start address of 2nd buffer used in continuous mode
CFG_SIZE	0x8	32	Sets the size of the buffers
CFG_STRIDE	0xC	32	Distance in byte between the end of a row and the beginning of the following
CFG_ROW_LEN	0x10	32	Length of the row in bytes
CFG_CURR_ADDR	0x14	32	Current address
CFG_BYTES_LEFT	0x18	32	Number of bytes left
CFG_CTRL	0x1C	32	2D generator control

CFG_SA_BUF0

Start address of 1st buffer used in legacy mode and continuous mode (HW double buffer)

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Start address of the 1st buffer (the 8 upper bits are forced to 0x1C)

CFG_SA_BUF1

Start address of 2nd buffer used in continuous mode

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Start address of the 2nd buffer, when in continuous mode (the 8 upper bits are forced to 0x1C)

CFG_SIZE

Sets the size of the buffers

Bit #	R/W	Name	Reset	Description
19:0	R/W	SIZE	0x0	Transfer size in bytes

CFG_STRIDE

Distance in byte between the end of a row and the beginning of the following

Bit #	R/W	Name	Reset	Description
19:0	R/W	STRIDE	0x0	Stride length in bytes

CFG_ROW_LEN

Length of the row in bytes

Bit #	R/W	Name	Reset	Description
19:0	R/W	ROW_LEN	0x0	Stride length in bytes

CFG_CURR_ADDR

Current address

Bit #	R/W	Name	Reset	Description
31:0	R	ADDRESS	0x0	Value of current address in the buffer (the 8 upper bits are always 0x1C)

CFG_BYTES_LEFT

Number of bytes left

Bit #	R/W	Name	Reset	Description
20:0	R	NUM_BYTES	0x0	Number of remaining bytes to transfer

CFG_CTRL

2D generator control

Bit #	R/W	Name	Reset	Description
0	R/W	CONT	0x0	Enable hardware double buffer support (continuous mode): b0: legacy mode; b1: continuous mode
1	R/W	EN	0x0	Write b1 to start a transfer or enqueue a new transfer if one is already running. On read: b0: no ongoing transfer; b1: ongoing transfer
4	W	STOP	0x0	Write b1 to stop the current transfer and reset the address generator

5.3.17 FIFO Channels

UDMA FIFO address generators

5.3.17.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CFG_SA_BUFFER	0x0	32	Start address for the FIFO
CFG_SIZE	0x8	32	Sets the size of the FIFO
CFG_EVT	0xC	32	Controls event generation
CFG_FIFO_FILL	0x18	32	Filling status of the FIFO
CFG_CTRL	0x1C	32	FIFO control

CFG_SA_BUFFER

Start address for the FIFO

Bit #	R/W	Name	Reset	Description
31:0	R/W	ADDRESS	0x0	Start address for the FIFO (the 8 upper bits are forced to 0x1C)

CFG_SIZE

Sets the size of the FIFO

Bit #	R/W	Name	Reset	Description
19:0	R/W	SIZE	0x10	FIFO size in bytes (minimum is 16, i.e. 0x10)

CFG_EVT

Controls event generation

Bit #	R/W	Name	Reset	Description
20:0	R/W	NUM_BYTES	0x0	Number of bytes to be received to trigger an event
31	R/W	EN	0x0	Enables event generation: b0: disabled; b1: enabled

CFG_FIFO_FILL

Filling status of the FIFO

Bit #	R/W	Name	Reset	Description
20:0	R	NUM_BYTES	0x0	Number of bytes in the FIFO

CFG_CTRL

FIFO control

Bit #	R/W	Name	Reset	Description
1	R/W	EN	0x0	Enable FIFO: b0: disabled; b1: enabled. If enabled, starts the FIFO on read
4	W	STOP	0x0	Disable FIFO: write b1 to stop the FIFO
8	R/W	TIMEOUT_MON	0x0	Select the port to monitor: b0: monitor RX port; b1: Monitor TX port

5.4 Cluster Peripherals

5.4.1 Cluster Controller

The cluster controller manages the following features:

- End of Computation status flag
- Configurable fetch activation for all cores of the Cluster
- Configurable core 0 boot address to define where to fetch first instruction in CL_CORE_0 after releasing the reset
- Configurable full cluster clock gating
- Configurable Cluster L1 memory arbitration policy
- Cluster cores resume command control
- Cluster cores halt status flags
- Configurable cluster cores debug halt command group mask policy

5.4.1.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
EOC	0x0	32	End Of Computation status register.
FETCH_EN	0x8	32	Cluster cores fetch enable configuration register.
CLOCK_GATE	0x20	32	Cluster clock gate configuration register.
DBG_RESUME	0x28	32	Cluster cores debug resume register.
DBG_HALT_STATUS	0x28	32	Cluster cores debug halt status register.
DBG_HALT_MASK	0x38	32	Cluster cores debug halt mask configuration register.
BOOT_ADDR0	0x40	32	Cluster core 0 boot address configuration register.
BOOT_ADDR1	0x44	32	Cluster core 1 boot address configuration register.
BOOT_ADDR2	0x48	32	Cluster core 2 boot address configuration register.
BOOT_ADDR3	0x4C	32	Cluster core 3 boot address configuration register.
BOOT_ADDR4	0x50	32	Cluster core 4 boot address configuration register.
BOOT_ADDR5	0x54	32	Cluster core 5 boot address configuration register.
BOOT_ADDR6	0x58	32	Cluster core 6 boot address configuration register.
BOOT_ADDR7	0x5C	32	Cluster core 7 boot address configuration register.
TCDM_ARB_POLICY_CH0	0x80	32	TCDM arbitration policy ch0 for cluster cores configuration register.
TCDM_ARB_POLICY_CH1	0x88	32	TCDM arbitration policy ch1 for DMA/HWCE configuration register.
TCDM_ARB_POLICY_CH0_REP	0xC0	32	Read only duplicate of TCDM_ARB_POLICY_CH0 register
TCDM_ARB_POLICY_CH1_REP	0xC8	32	Read only duplicate of TCDM_ARB_POLICY_CH1 register

EOC

End Of Computation status register.

Bit #	R/W	Name	Reset	Description
0	R/W	EOC	0x0	End of computation status flag: b0: program execution ongoing; b1: end of computation reached

FETCH_EN

Cluster cores fetch enable configuration register.

Bit #	R/W	Name	Reset	Description
0	R/W	CORE0	0x0	Core 0 fetch enable: b0: disabled; b1: enabled
1	R/W	CORE1	0x0	Core 1 fetch enable: b0: disabled; b1: enabled
2	R/W	CORE2	0x0	Core 2 fetch enable: b0: disabled; b1: enabled
3	R/W	CORE3	0x0	Core 3 fetch enable: b0: disabled; b1: enabled
4	R/W	CORE4	0x0	Core 4 fetch enable: b0: disabled; b1: enabled
5	R/W	CORE5	0x0	Core 5 fetch enable: b0: disabled; b1: enabled
6	R/W	CORE6	0x0	Core 6 fetch enable: b0: disabled; b1: enabled
7	R/W	CORE7	0x0	Core 7 fetch enable: b0: disabled; b1: enabled

CLOCK_GATE

Cluster clock gate configuration register.

Bit #	R/W	Name	Reset	Description
0	R/W	EN	0x0	Cluster clock gate: b0: disabled; b1: enabled

DBG_RESUME

Cluster cores debug resume register.

Bit #	R/W	Name	Reset	Description
0	W	CORE0	0x0	Core 0 debug resume: b0: stay halted; b1: resume execution on core 0
1	W	CORE1	0x0	Core 1 debug resume: b0: stay halted; b1: resume execution on core 1
2	W	CORE2	0x0	Core 2 debug resume: b0: stay halted; b1: resume execution on core 2
3	W	CORE3	0x0	Core 3 debug resume: b0: stay halted; b1: resume execution on core 3
4	W	CORE4	0x0	Core 4 debug resume: b0: stay halted; b1: resume execution on core 4
5	W	CORE5	0x0	Core 5 debug resume: b0: stay halted; b1: resume execution on core 5
6	W	CORE6	0x0	Core 6 debug resume: b0: stay halted; b1: resume execution on core 6
7	W	CORE7	0x0	Core 7 debug resume: b0: stay halted; b1: resume execution on core 7

DBG_HALT_STATUS

Cluster cores debug halt status register.

Bit #	R/W	Name	Reset	Description
0	R	CORE0	0x0	Core 0 debug halt status: b0: running; b1: halted
1	R	CORE1	0x0	Core 1 debug halt status: b0: running; b1: halted
2	R	CORE2	0x0	Core 2 debug halt status: b0: running; b1: halted
3	R	CORE3	0x0	Core 3 debug halt status: b0: running; b1: halted
4	R	CORE4	0x0	Core 4 debug halt status: b0: running; b1: halted
5	R	CORE5	0x0	Core 5 debug halt status: b0: running; b1: halted
6	R	CORE6	0x0	Core 6 debug halt status: b0: running; b1: halted
7	R	CORE7	0x0	Core 7 debug halt status: b0: running; b1: halted

DBG_HALT_MASK

Cluster cores debug halt mask configuration register.

Bit #	R/W	Name	Reset	Description
0	R/W	CORE0	0x0	Core 0 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
1	R/W	CORE1	0x0	Core 1 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
2	R/W	CORE2	0x0	Core 2 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
3	R/W	CORE3	0x0	Core 3 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
4	R/W	CORE4	0x0	Core 4 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
5	R/W	CORE5	0x0	Core 5 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
6	R/W	CORE6	0x0	Core 6 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops
7	R/W	CORE7	0x0	Core 7 debug halt mask: when set, the core is part of the mask group and stops when one of the members of the group stops

BOOT_ADDR0

Cluster core 0 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR1

Cluster core 1 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR2

Cluster core 2 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR3

Cluster core 3 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR4

Cluster core 4 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR5

Cluster core 5 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR6

Cluster core 6 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

BOOT_ADDR7

Cluster core 7 boot address configuration register.

Bit #	R/W	Name	Reset	Description
31:0	R/W	BA	0x0	Boot address for corresponding cluster core

TCDM_ARB_POLICY_CH0

TCDM arbitration policy ch0 for cluster cores configuration register.

Bit #	R/W	Name	Reset	Description
0	R/W	POL	0x0	TCDM arbitration policy for cluster cores: b0: fair round robin; b1: fixed order

TCDM_ARB_POLICY_CH1

TCDM arbitration policy ch1 for DMA/HWCE configuration register.

Bit #	R/W	Name	Reset	Description
0	R/W	POL	0x0	TCDM arbitration policy for DMA and NE16: b0: fair round robin; b1: fixed order

TCDM_ARB_POLICY_CH0_REP

Read only duplicate of TCDM_ARB_POLICY_CH0 register

Bit #	R/W	Name	Reset	Description
0	R	POL	0x0	TCDM arbitration policy for cluster cores: b0: fair round robin; b1: fixed order

TCDM_ARB_POLICY_CH1_REP

Read only duplicate of TCDM_ARB_POLICY_CH1 register

Bit #	R/W	Name	Reset	Description
0	R	POL	0x0	TCDM arbitration policy for DMA and NE16: b0: fair round robin; b1: fixed order

5.4.2 Cluster Event Unit

5.4.2.1 Description

The cluster event unit manages hardware and software events and interrupts for cluster cores, both for events generated internally or coming from external sources inside or outside of the cluster. Events and interrupts configuration use a 32-bit mapping of the different events according to the table below:

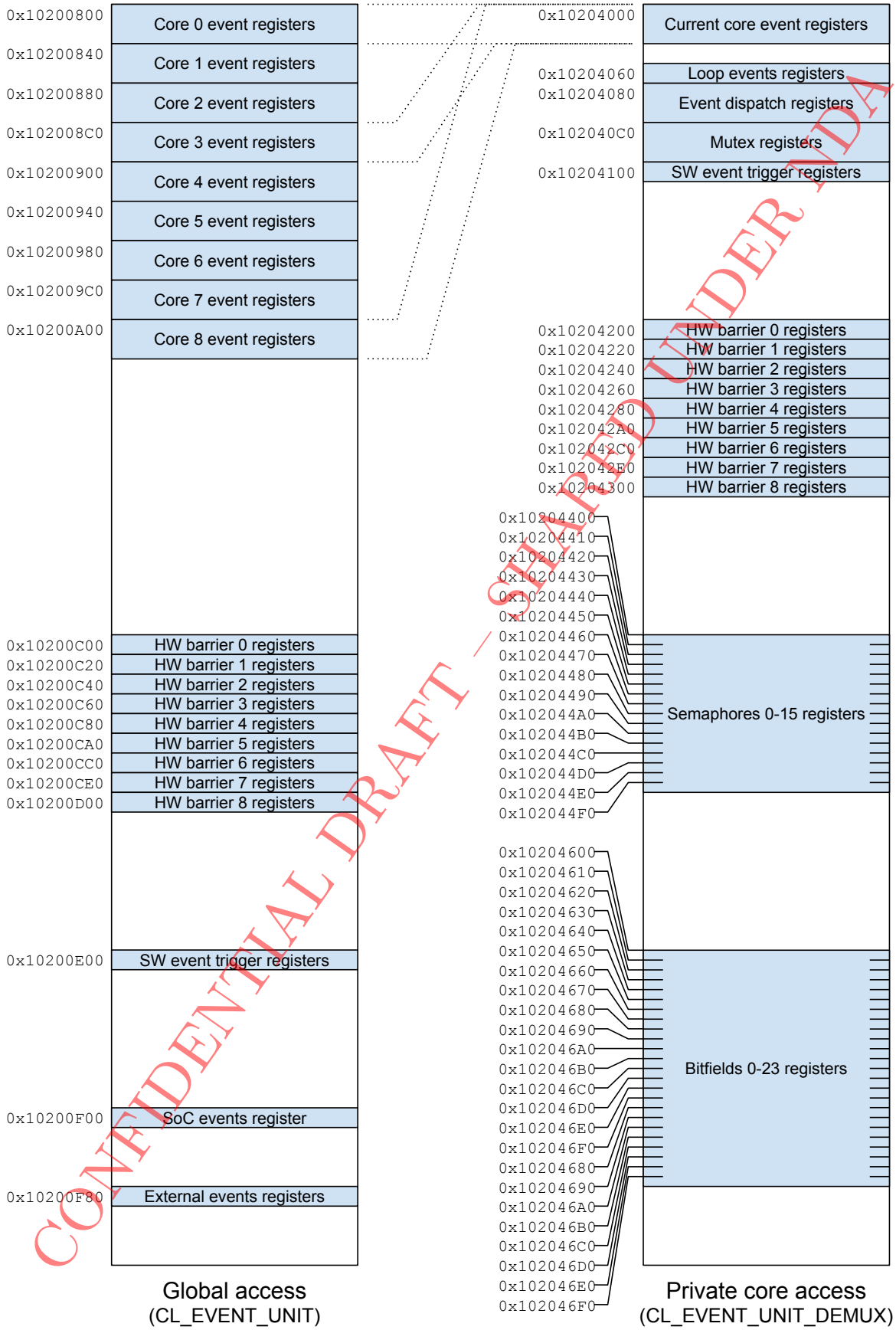
Event bit	Source event
0-7	Software events (8 independent events)
8-9	Cluster DMA events
10-11	Timer events (hi & lo)
12-15	NE16 events
16	Barrier event for current core
17	Mutex event for current core
18	Dispatched event for current core
24	Compressor/decompressor event
27	Event from SoC events FIFO

The cluster event unit also offers support for HW barriers, semaphores and bitfields.

There are two ways to access the cluster event unit registers:

- Through the CL_EVENT_UNIT address space, which shows a global address space for all 9 cluster cores. Note that some registers (e.g. related to semaphores and bitfields) are private to cluster cores and are not accessible through this address space.
- From any cluster core, through the CL_EVENT_UNIT_DEMUX address space, which gives a direct access to the registers, including semaphores and bitfields. Note that EU_CORE_* registers of other cluster cores cannot be accessed through this address space.

The following figure details those address spaces. Refer to it to determine the base address to be used to apply the proper offset for each of the different subsets of registers described below.



5.4.2.2 Register map for core events

Overview

Name	Offset	Width	Description
EU_CORE_MASK	0x0	32	Core event mask register
EU_CORE_MASK_AND	0x4	32	Clear bits of core event mask register
EU_CORE_MASK_OR	0x8	32	Set bits of core event mask register
EU_CORE_MASK_IRQ	0xC	32	Core intrerrupt mask register
EU_CORE_MASK_IRQ_AND	0x10	32	Clear bits of core intrerrupt mask register
EU_CORE_MASK_IRQ_OR	0x14	32	Set bits of core intrerrupt mask register
EU_CORE_STATUS	0x18	32	Core status register
EU_CORE_BUFFER	0x1C	32	Event status register
EU_CORE_BUFFER_MASKED	0x20	32	Event status register with event mask applied
EU_CORE_BUFFER_IRQ_MASKED	0x24	32	Event status register with interrupt mask applied
EU_CORE_BUFFER_CLEAR	0x28	32	Clear bits of event status register
EU_CORE_SW_EVENTS_MASK	0x2C	32	Target core mask for SW wait events
EU_CORE_SW_EVENTS_MASK_AND	0x30	32	Clear bits of target core mask for SW wait events
EU_CORE_SW_EVENTS_MASK_OR	0x34	32	Set bits of target core mask for SW wait events
EU_CORE_EVENT_WAIT	0x38	32	Wait for event (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_EVENT_WAIT_CLEAR	0x3C	32	Wait for event and clear it (only accessible from CL_EVENT_UNIT_DEMUX)

EU_CORE_MASK

Core event mask register

Bit #	R/W	Name	Reset	Description
31:0	R/W	EVENT_MASK	0x00000000	Mask bits for events: set a bit to b1 to enable corresponding event as a wake-up source for the core

EU_CORE_MASK_AND

Clear bits of core event mask register

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_CLR	0x00000000	Write b1 to 1 or more bits to clear the corresponding bits of EU_CORE_MASK

EU_CORE_MASK_OR

Set bits of core event mask register

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_SET	0x00000000	Write b1 to 1 or more bits to set the corresponding bits of EU_CORE_MASK

EU_CORE_MASK_IRQ

Core intrerrupt mask register

Bit #	R/W	Name	Reset	Description
31:0	R/W	EVENT_MASK	0x00000000	Mask bits for events: set a bit to b1 to enable corresponding event as an interrupt source for the core

EU_CORE_MASK_IRQ_AND

Clear bits of core intrerrupt mask register

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_CLR	0x00000000	Write b1 to 1 or more bits to clear the corresponding bits of EU_CORE_MASK_IRQ

EU_CORE_MASK_IRQ_OR

Set bits of core intrerrupt mask register

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_SET	0x00000000	Write b1 to 1 or more bits to set the corresponding bits of EU_CORE_MASK_IRQ

EU_CORE_STATUS

Core status register

Bit #	R/W	Name	Reset	Description
0	R	CLK_STATUS	0x0	Status of core clock: b0: clock is stopped; b1: clock is running

EU_CORE_BUFFER

Event status register

Bit #	R/W	Name	Reset	Description
31:0	R	EVENTS	0x00000000	Status bits for events: a bit reads as b1 when the corresponding event has occurred at least once since this bit was cleared

EU_CORE_BUFFER_MASKED

Event status register with event mask applied

Bit #	R/W	Name	Reset	Description
31:0	R	MASKED_EVT	0x00000000	Holds the value of EU_CORE_BUFFER masked by EU_CORE_MASK value

EU_CORE_BUFFER_IRQ_MASKED

Event status register with interrupt mask applied

Bit #	R/W	Name	Reset	Description
31:0	R	MASKED_EVT	0x00000000	Holds the value of EU_CORE_BUFFER masked by EU_CORE_MASK_IRQ value

EU_CORE_BUFFER_CLEAR

Clear bits of event status register

Bit #	R/W	Name	Reset	Description
31:0	W	EVT_CLR	0x00000000	Write b1 to one or more bits to clear the corresponding bits in EU_CORE_BUFFER. Note that an event occuring at the same time may be missed

EU_CORE_SW_EVENTS_MASK

Target core mask for SW wait events

Bit #	R/W	Name	Reset	Description
7:0	R/W	CL_INTERN_EVT	0x0000	Mask bits used when a SW event is triggered from EU_CORE_TRIGG_SW_EVENT_WAIT or EU_CORE_TRIGG_SW_EVENT_WAIT_CLEAR. Note that for no-wait SW event triggering, the used mask is set independently when writting the EU_CORE_TRIGG_SW_EVENT register

EU_CORE_SW_EVENTS_MASK_AND

Clear bits of target core mask for SW wait events

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_CLR	0x00000000	Write b1 to 1 or more bits to clear the corresponding bits of EU_CORE_SW_EVENTS_MASK

EU_CORE_SW_EVENTS_MASK_OR

Set bits of target core mask for SW wait events

Bit #	R/W	Name	Reset	Description
31:0	W	MASK_SET	0x00000000	Write b1 to 1 or more bits to set the corresponding bits of EU_CORE_SW_EVENTS_MASK

EU_CORE_EVENT_WAIT

Wait for event (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R	EVENT	0x00000000	Reading this register stops the clock of the core until at least one event with the corresponding mask bit set to 1 occurs. The returned value is identical to that of EU_CORE_BUFFER_MASKED

EU_CORE_EVENT_WAIT_CLEAR

Wait for event and clear it (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R	EVENT	0x00000000	Reading this register has the same behavior as reading EU_CORE_EVENT_WAIT, except that bits of EU_CORE_BUFFER that are set to 1 in EU_CORE_MASK are cleared to 0 after the read

5.4.2.3 Register map for mutex events

Overview

Name	Offset	Width	Description
EU_CORE_HW_MUTEX0	0x0	32	Access to HW mutex 0 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX1	0x4	32	Access to HW mutex 1 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX2	0x8	32	Access to HW mutex 2 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX3	0xC	32	Access to HW mutex 3 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX4	0x10	32	Access to HW mutex 4 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX5	0x14	32	Access to HW mutex 5 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX6	0x18	32	Access to HW mutex 6 (only accessible from CL_EVENT_UNIT_DEMUX)
EU_CORE_HW_MUTEX7	0x1C	32	Access to HW mutex 7 (only accessible from CL_EVENT_UNIT_DEMUX)

EU_CORE_HW_MUTEX0

Access to HW mutex 0 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Reading tries to lock the corresponding mutex, writing unlocks it. The value written during an unlock is transmitted to the core which locks the mutex next. Note that there is no protection against illegal unlocks in hardware, i.e. the runtime SW is responsible for making sure that only the core that has locked a mutex performs a write on this register.

EU_CORE_HW_MUTEX1

Access to HW mutex 1 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX2

Access to HW mutex 2 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX3

Access to HW mutex 3 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX4

Access to HW mutex 4 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX5

Access to HW mutex 5 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX6

Access to HW mutex 6 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

EU_CORE_HW_MUTEX7

Access to HW mutex 7 (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R/W	VALUE	0x00000000	Same behavior as HW mutex 0.

5.4.2.4 Register map for software events

Overview

Name	Offset	Width	Description
EU_CORE_TRIGG_SW_EVENT0	0x0	32	Trigger SW event 0
EU_CORE_TRIGG_SW_EVENT1	0x4	32	Trigger SW event 1
EU_CORE_TRIGG_SW_EVENT2	0x8	32	Trigger SW event 2
EU_CORE_TRIGG_SW_EVENT3	0xC	32	Trigger SW event 3
EU_CORE_TRIGG_SW_EVENT4	0x10	32	Trigger SW event 4
EU_CORE_TRIGG_SW_EVENT5	0x14	32	Trigger SW event 5
EU_CORE_TRIGG_SW_EVENT6	0x18	32	Trigger SW event 6
EU_CORE_TRIGG_SW_EVENT7	0x1C	32	Trigger SW event 7
EU_CORE_TRIGG_SW_EVENT0_WAIT	0x40	32	Trigger SW event 0 and go to sleep
EU_CORE_TRIGG_SW_EVENT1_WAIT	0x44	32	Trigger SW event 1 and go to sleep
EU_CORE_TRIGG_SW_EVENT2_WAIT	0x48	32	Trigger SW event 2 and go to sleep
EU_CORE_TRIGG_SW_EVENT3_WAIT	0x4C	32	Trigger SW event 3 and go to sleep
EU_CORE_TRIGG_SW_EVENT4_WAIT	0x40	32	Trigger SW event 4 and go to sleep
EU_CORE_TRIGG_SW_EVENT5_WAIT	0x44	32	Trigger SW event 5 and go to sleep
EU_CORE_TRIGG_SW_EVENT6_WAIT	0x48	32	Trigger SW event 6 and go to sleep
EU_CORE_TRIGG_SW_EVENT7_WAIT	0x4C	32	Trigger SW event 7 and go to sleep
EU_CORE_TRIGG_SW_EVENT0_WAIT_CLEAR	0x80	32	Trigger SW event 0, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT1_WAIT_CLEAR	0x84	32	Trigger SW event 1, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT2_WAIT_CLEAR	0x88	32	Trigger SW event 2, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT3_WAIT_CLEAR	0x8C	32	Trigger SW event 3, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT4_WAIT_CLEAR	0x80	32	Trigger SW event 4, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT5_WAIT_CLEAR	0x84	32	Trigger SW event 5, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT6_WAIT_CLEAR	0x88	32	Trigger SW event 6, go to sleep and clear on wake-up
EU_CORE_TRIGG_SW_EVENT7_WAIT_CLEAR	0x8C	32	Trigger SW event 7, go to sleep and clear on wake-up

EU_CORE_TRIGG_SW_EVENT0

Trigger SW event 0

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Writing to this register triggers software event with ID 0 for all cores with corresponding bit written as b1

EU_CORE_TRIGG_SW_EVENT1

Trigger SW event 1

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT2

Trigger SW event 2

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT3

Trigger SW event 3

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT4

Trigger SW event 4

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT5

Trigger SW event 5

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT6

Trigger SW event 6

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT7

Trigger SW event 7

Bit #	R/W	Name	Reset	Description
8:0	W	EVT_MASK	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0, except for the event ID used

EU_CORE_TRIGG_SW_EVENT0_WAIT

Trigger SW event 0 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Reading from this register triggers software event with ID 0 for all cores with corresponding bit written as b1 in EU_CORE_SW_EVENTS_MASK register. Current core then goes to sleep with the same behavior as when reading EU_CORE_-EVENT_WAIT

EU_CORE_TRIGG_SW_EVENT1_WAIT

Trigger SW event 1 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT2_WAIT

Trigger SW event 2 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT3_WAIT

Trigger SW event 3 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT4_WAIT

Trigger SW event 4 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT5_WAIT

Trigger SW event 5 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT6_WAIT

Trigger SW event 6 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT7_WAIT

Trigger SW event 7 and go to sleep

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_EVENT0_WAIT, except for the event ID used

EU_CORE_TRIGG_SW_EVENT0_WAIT_CLEAR

Trigger SW event 0, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Reading from this register triggers software event with ID 0 for all cores with corresponding bit written as b1 in EU_CORE_SW_EVENTS_MASK register. Current core then goes to sleep with the same behavior as when reading EU_CORE_EVENT_WAIT_CLEAR

EU_CORE_TRIGG_SW_EVENT1_WAIT_CLEAR

Trigger SW event 1, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT2_WAIT_CLEAR

Trigger SW event 2, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT3_WAIT_CLEAR

Trigger SW event 3, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT4_WAIT_CLEAR

Trigger SW event 4, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT5_WAIT_CLEAR

Trigger SW event 5, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT6_WAIT_CLEAR

Trigger SW event 6, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT_CLEAR, except for the event ID used

EU_CORE_TRIGG_SW_EVENT7_WAIT_CLEAR

Trigger SW event 7, go to sleep and clear on wake-up

Bit #	R/W	Name	Reset	Description
8:0	W	EVENT	0x000	Same behavior as EU_CORE_TRIGG_SW_-EVENT0_WAIT_CLEAR, except for the event ID used

5.4.2.5 Register map for SoC events

Overview

Name	Offset	Width	Description
EU_CORE_CUR-RENT_EVENT	0x0	32	Access to SoC event FIFO (only accessible from CL_EVENT_-UNIT)

EU_CORE_CURRENT_EVENT

Access to SoC event FIFO (only accessible from CL_EVENT_UNIT)

Bit #	R/W	Name	Reset	Description
7:0	R	EVENT_ID	0x00	Oldest SoC peripheral event ID that was written into the FIFO. After a read of this register, the next event ID (if any) will be read at the next access
31	R	VALID	0x0	Valid bit: if read as b1, the EVENT_ID field is valid

5.4.2.6 Register map for hardware barriers

Overview

Name	Offset	Width	Description
HW_BARR_TRIGGER_MASK	0x0	32	Trigger mask for HW barrier
HW_BARR_STATUS	0x4	32	Current status of HW barrier
HW_BARR_TARGET_MASK	0xC	32	Barrier target mask
HW_BARR_TRIGGER	0x10	32	Trigger the HW barrier
HW_BARR_TRIGGER_SELF	0x14	32	Self-trigger the HW barrier (only accessible from CL_EVENT_UNIT_DEMUX)
HW_BARR_TRIGGER_WAIT	0x18	32	Self-trigger and go to sleep (only accessible from CL_EVENT_UNIT_DEMUX)
HW_BARR_TRIGGER_WAIT_CLEAR	0x1C	32	Self-trigger, go to sleep and clear on wake-up (only accessible from CL_EVENT_UNIT_DEMUX)

HW_BARR_TRIGGER_MASK

Trigger mask for HW barrier

Bit #	R/W	Name	Reset	Description
8:0	R/W	MASK	0x000	Trigger mask for barrier to match: every bit set to b1 in MASK must be set in HW_BARR_STATUS for the barrier to be triggered. Writing all bits to b0 disables the HW barrier

HW_BARR_STATUS

Current status of HW barrier

Bit #	R/W	Name	Reset	Description
8:0	R	STATUS	0x000	Current status of the barrier: a bit reading as b1 means that the corresponding bit has already been triggered. This field is cleared to 0x000 when the barrier triggers, i.e. when status matches the mask value in HW_BARR_TRIGGER_MASK

HW_BARR_TARGET_MASK

Barrier target mask

Bit #	R/W	Name	Reset	Description
8:0	R	CORE_MASK	0x000	Target cores mask: once the HW barrier triggers, i.e. when HW_BARR_STATUS matches HW_BARR_TRIGGER_MASK, the event line corresponding to this barrier is asserted for all cores which have the corresponding bit set to b1 in this field

HW_BARR_TRIGGER

Trigger the HW barrier

Bit #	R/W	Name	Reset	Description
8:0	R	VALUE	0x000	Every bit written as b1 triggers the corresponding bit of HW barrier, i.e. the corresponding bit is set in HW_BARR_STATUS

HW_BARR_TRIGGER_SELF

Self-trigger the HW barrier (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R	DUMMY	0x00000000	When read, the bit corresponding to current cluster core is triggered in the barrier, i.e. the corresponding bit is set in HW_BARR_STATUS

HW_BARR_TRIGGER_WAIT

Self-trigger and go to sleep (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R	EVENT	0x00000000	Same behavior as HW_BARR_TRIGGER_SELF, but the core then goes to sleep. On wakeup, the read data is identical to the content of the EU_CORE_BUFFER_MASKED register of the issuing core

HW_BARR_TRIGGER_WAIT_CLEAR

Self-trigger, go to sleep and clear on wake-up (only accessible from CL_EVENT_UNIT_DEMUX)

Bit #	R/W	Name	Reset	Description
31:0	R	EVENT	0x00000000	Same behavior as HW_BARR_TRIGGER_WAIT, except that bits of EU_CORE_BUFFER that are set to 1 in EU_CORE_MASK are cleared to 0 after the read

5.4.2.7 Register map for semaphores

Overview

Name	Offset	Width	Description
VALUE	0x0	32	Set or get the value of the semaphore
COUNTER	0x4	32	Atomically increase or decrease the value of the semaphore
COUNTER2	0x8	32	Read and increase the value of the semaphore

VALUE

Set or get the value of the semaphore

Bit #	R/W	Name	Reset	Description
11:0	R/W	COUNT	0x000	Current value of semaphore counter

COUNTER

Atomically increase or decrease the value of the semaphore

Bit #	R/W	Name	Reset	Description
11:0	R/W	INC_DEC	0x000	Writing to this register increases the value of the semaphore by the value written; reading this register returns current value and decrements it, atomically

COUNTER2

Read and increase the value of the semaphore

Bit #	R/W	Name	Reset	Description
11:0	R	COUNT_INC	0x000	Reading this register returns current value of the semaphore and increments it, atomically

5.4.2.8 Register map for bitfields

Overview

Name	Offset	Width	Description
VALUE	0x0	32	Set or get the value of the bitfield
SET	0x4	32	Selectively set bitfield bits
CLEAR	0x8	32	Selectively clear bitfield bits
ALLOC	0xC	32	Get position and clear least significant set bit

VALUE

Set or get the value of the bitfield

Bit #	R/W	Name	Reset	Description
31:0	R/W	BITS	0x00000000	Current value of the bitfield

SET

Selectively set bitfield bits

Bit #	R/W	Name	Reset	Description
31:0	W	SET_MASK	0x00000000	Write b1 to 1 or more bits to set the corresponding bits of the bitfield

CLEAR

Selectively clear bitfield bits

Bit #	R/W	Name	Reset	Description
31:0	W	CLEAR_MASK	0x00000000	Write b1 to 1 or more bits to clear the corresponding bits of the bitfield

ALLOC

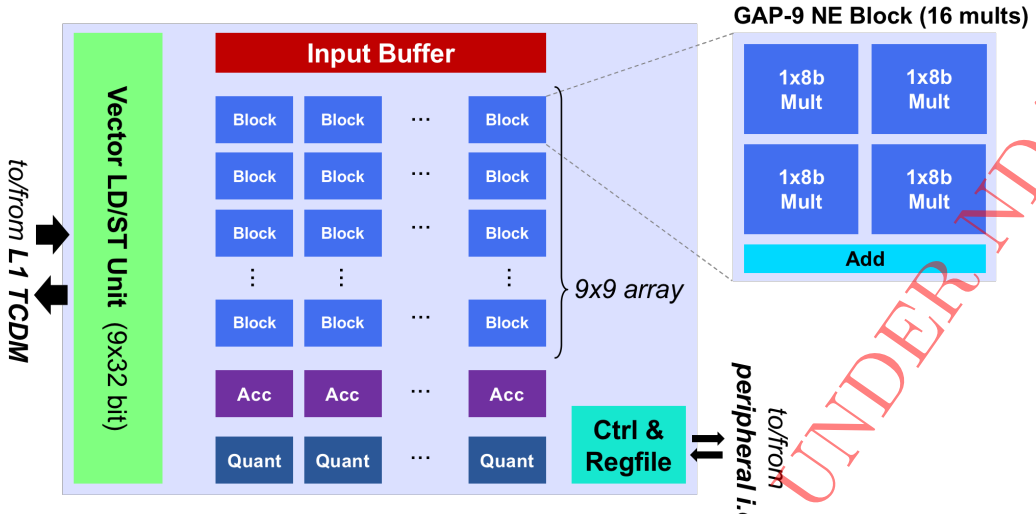
Get position and clear least significant set bit

Bit #	R/W	Name	Reset	Description
31:0	R	FIRST_ONE	0x00000000	Reading this field returns the position (from 0 to 31) of the least significant bit set at 1, and clears it to 0. If no bit is set, the read value is 32

5.4.3 NE16

The Neural Engine 16 (NE16) is an accelerator specialized in the execution of Deep Learning inference kernels such as convolutional, depthwise and fully connected kernels. It is based around a few basic ideas:

- HWC layout for activations and consequently computation based on Matrix-Vector innermost loops.
- Bit-Serial computation over the bit weights dimension, to seamlessly support from 2 to 8 bits of weights, including "unusual" bitwidths such as 3, 5, 6, 7 bits.
- Output-stationary dataflow with tiling in spatial dimensions to partially exploit spatial data reuse for 3×3 filters.
- Layout of weight data centered on memory access efficiency, and therefore different across different operating modes.



The NE16 supports four alternative operating modes:

- 3×3 mode: the NE16 is configured and optimized to run 3×3 convolutional layers.
- 3×3 depthwise mode: the NE16 is configured and optimized to run 3×3 depthwise layers.
- 1×1 mode: the NE16 is configured and optimized to run 1×1 pointwise layers.
- Linear mode: the NE16 is configured and optimized to run Matrix-Vector products.

Orthogonally, the user can set up the NE16 to operate on 8-bit input and output tensors (basic mode) or on 16-bit input and output tensors (mode16 mode).

More details on NE16 operations and programming are given in a separate document. Contact Greenwaves Technologies for more information.

5.4.3.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
TRIGGER	0x0	32	Commit/trigger jobs
ACQUIRE	0x4	32	Offload jobs
STATUS	0xC	32	Status of jobs
RUNNING_JOB	0x10	32	Running job ID
SOFT_CLEAR	0x14	32	Soft reset of NE16 state
WEIGHTS_PTR	0x20	32	Pointer to weights tensor
INFEAT_PTR	0x24	32	Pointer to InFeat tensor
OUTFEAT_PTR	0x28	32	Pointer to OutFeat tensor
SCALE_PTR	0x2C	32	Pointer to ScaleScaleN parameters
SCALE_SHIFT_PTR	0x30	32	Pointer to ScaleShift parameters
SCALE_BIAS_PTR	0x34	32	Pointer to ScaleBias parameters
INFEAT_D0_STRIDE	0x38	32	InFeat tensor d0 stride
INFEAT_D1_STRIDE	0x3C	32	InFeat tensor d1 stride
INFEAT_D2_STRIDE	0x40	32	InFeat tensor d2 stride
OUTFEAT_D0_STRIDE	0x44	32	OutFeat tensor d0 stride
OUTFEAT_D1_STRIDE	0x48	32	OutFeat tensor d1 stride
OUTFEAT_D2_STRIDE	0x4C	32	OutFeat tensor d2 stride
WEIGHTS_D0_STRIDE	0x50	32	Weights tensor d0 stride
WEIGHTS_D1_STRIDE	0x54	32	Weights tensor d1 stride
WEIGHTS_D2_STRIDE	0x58	32	Weights tensor d2 stride
SUBTILE_REM0	0x5C	32	Subtile remainders 0
SUBTILE_REM1	0x60	32	Subtile remainders 1
SUBTILE_REM2	0x64	32	Subtile remainders 2
SUBTILE_NB0	0x68	32	Subtile numbers 0
SUBTILE_NB1	0x6C	32	Subtile numbers 1
PADDING	0x70	32	Padding
WEIGHT_OFFSET	0x74	32	Weight offset
FILTER_MASK	0x78	32	Filter masking
CONFIG0	0x7C	32	Main configuration register

TRIGGER

Commit/trigger jobs

Bit #	R/W	Name	Reset	Description
31:0	W	TRIG_CMD	0x0	Write 0 to commit a job, unlock controller and start execution. Write a non-0 value to commit a job and unlock controller without starting execution, which will be started when the next job is committed and triggered. Only the first job in a queue may be committed without triggering it.

ACQUIRE

Offload jobs

Bit #	R/W	Name	Reset	Description
31:0	R	JOB_ID	0x0	Read to start a job offload and lock controller. Returns job ID.

STATUS

Status of jobs

Bit #	R/W	Name	Reset	Description
0	R	STATUS0	0x0	Status of job 0: b1 if job 0 is currently enqueued or running, b0 otherwise
8	R	STATUS1	0x0	Status of job 1: b1 if job 1 is currently enqueued or running, b0 otherwise

RUNNING_JOB

Running job ID

Bit #	R/W	Name	Reset	Description
31:0	R	JOB_ID	0x0	ID of currently running, if any job is running; otherwise, ID of the last job that has been run

SOFT_CLEAR

Soft reset of NE16 state

Bit #	R/W	Name	Reset	Description
31:0	W	CLR_CMD	0x0	Write 0 to clear the full status of the accelerator IP, including the register file; write a non-0 value to clear the status of the accelerator IP, except for the register file

WEIGHTS_PTR

Pointer to weights tensor

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to weights tensor in memory (d3=Ko, d2=KiMaj, d1=Qw, d0=Fx×Fy×Kimin for 3×3 modes; d2=Ko, d1=KiMaj, d0=Qw×KiMin for 1×1 mode; d2=Ko, d1=Qw, d0=KiMaj×KiMin for linear mode).

INFEAT_PTR

Pointer to InFeat tensor

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to InFeat tensor in memory (d2=Hi, d1=Wi, d0=Ki)

OUTFEAT_PTR

Pointer to OutFeat tensor

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to OutFeat tensor in memory (d2=Ho, d1=Wo, d0=Ko)

SCALE_PTR

Pointer to ScaleScaleN parameters

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to Scale/ScaleN parameters in memory (d0=Ko)

SCALE_SHIFT_PTR

Pointer to ScaleShift parameters

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to ScaleShift parameters in memory (d0=Ko)

SCALE_BIAS_PTR

Pointer to ScaleBias parameters

Bit #	R/W	Name	Reset	Description
31:0	R/W	POINTER	0x0	Pointer to ScaleBias parameters in memory (d0=Ko)

INFEAT_D0_STRIDE

InFeat tensor d0 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

INFEAT_D1_STRIDE

InFeat tensor d1 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

INFEAT_D2_STRIDE

InFeat tensor d2 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

OUTFEAT_D0_STRIDE

OutFeat tensor d0 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

OUTFEAT_D1_STRIDE

OutFeat tensor d1 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

OUTFEAT_D2_STRIDE

OutFeat tensor d2 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

WEIGHTS_D0_STRIDE

Weights tensor d0 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

WEIGHTS_D1_STRIDE

Weights tensor d1 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

WEIGHTS_D2_STRIDE

Weights tensor d2 stride

Bit #	R/W	Name	Reset	Description
31:0	R/W	STRIDE	0x0	Stride value

SUBTILE_REM0

Subtile remainders 0

Bit #	R/W	Name	Reset	Description
15:0	R/W	KI	0x0	Ki remainder
31:16	R/W	KO	0x0	Ko remainder

SUBTILE_REM1

Subtile remainders 1

Bit #	R/W	Name	Reset	Description
15:0	R/W	WO	0x0	Wo remainder
31:16	R/W	HO	0x0	Ho remainder

SUBTILE_REM2

Subtile remainders 2

Bit #	R/W	Name	Reset	Description
15:0	R/W	WI	0x0	Wi remainder
31:16	R/W	HI	0x0	Hi remainder

SUBTILE_NB0

Subtile numbers 0

Bit #	R/W	Name	Reset	Description
15:0	R/W	KI	0x0	Ki remainder
31:16	R/W	KO	0x0	Ko remainder

SUBTILE_NB1

Subtile numbers 1

Bit #	R/W	Name	Reset	Description
15:0	R/W	WO	0x0	Wo remainder
31:16	R/W	HO	0x0	Ho remainder

PADDING

Padding

Bit #	R/W	Name	Reset	Description
15:0	R/W	VALUE	0x0	Padding value
19:16	R/W	LEFT	0x0	Number of spatially padded pixels in the left subtile border
23:20	R/W	BOTTOM	0x0	Number of spatially padded pixels in the bottom subtile border (counted from 5 pixels upward)
27:24	R/W	RIGHT	0x0	Number of spatially padded pixels in the right subtile border (counted from 5 pixels leftward)
31:28	R/W	TOP	0x0	Number of spatially padded pixels in the top subtile border

WEIGHT_OFFSET

Weight offset

Bit #	R/W	Name	Reset	Description
7:0	R/W	VALUE	0x0	Value of weights offset

FILTER_MASK

Filter masking

Bit #	R/W	Name	Reset	Description
7:0	R/W	LEFT	0x0	Left mask
15:8	R/W	BOTTOM	0x0	Bottom mask
23:16	R/W	RIGHT	0x0	Right mask
31:24	R/W	TOP	0x0	Top mask

CONFIG0

Main configuration register

Bit #	R/W	Name	Reset	Description
2:0	R/W	QWM1	0x0	Weight bits minus 1
3	R/W	MODE16	0x0	Enable mode16: b0: 8-bit mode; b1: 16-bit mode
4	R/W	STREAMOUT	0x0	Normalization/quantization before streamout: b0: streamout only; b1: quantization+streamout
6:5	R/W	FILTER_MODE	0x0	Filter mode: b00=3×3; b01=3×3 depthwise; b10=1×1; b11: reserved
7	R/W	LINEAR	0x0	Linear mode: b0: normal operation; b1: linear mode
8	R/W	STRIDED_2X2	0x0	Strided 2×2 mode: b0: normal operation; b1: strided mode
13:12	R/W	NORM_BITS	0x0	Normalization bits: b00: 8 bits; b01: 16 bits; b10: 32b; b11: reserved
14	R/W	STREAMIN	0x0	Streamin mode: b0: normal operation; b1: enable streamin
15	R/W	WEIGHT_OFFS	0x0	Weight offset configuration: b0: symmetric weights; b1: use layer-wise weight offset
20:16	R/W	RIGHT_SHIFT	0x0	Quantization right shift
22:21	R/W	QUANT_BITS	0x0	Quantization bits: b00: 8 bits; b01: 16 bits; b10: 32b; b11: reserved
23	R/W	QUANT_RECT	0x0	Quantization recttify: b0: rectify, consider as unsigned; b1: do not rectify, keep sign
24	R/W	NORM_SHIFT	0x0	Norm option shift: b0: use quantization right shift; b1: load with norm
25	R/W	NORM_BIAS	0x0	Norm option bias: b0: do not load bias; b1: load bias

5.4.4 Cluster I-Cache Controller

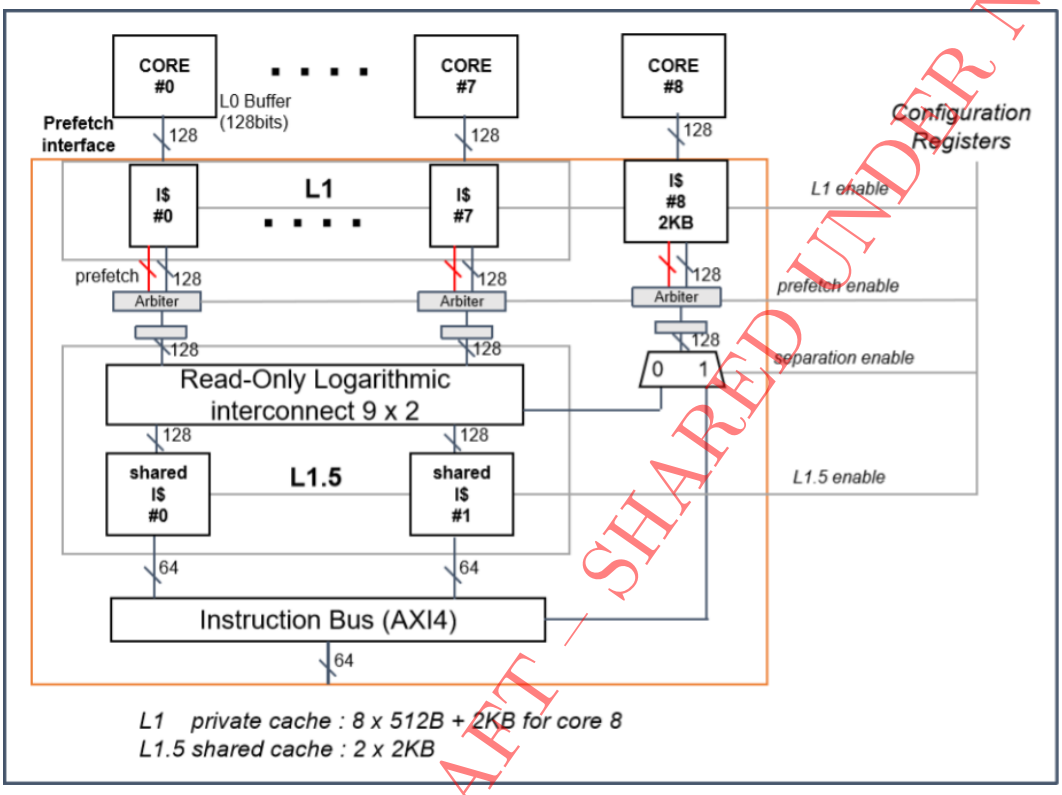
The cluster hierarchical instruction cache controller is split into 9 private banks – one for each cluster core – and 2 shared banks. It offers the following features:

- Bypassable instruction cache
- Flush and selective flush commands

5.4.4.1 Cache Operation

The hierarchical instruction cache is optimized to perform efficient instruction caching for the multiple cores of the cluster. It fetches the code from the L2 memory and delivers data to the cores with the granularity of a cache line, i.e. 128 bits.

The figure below shows a top-level view of the cache.



The cache is organized in a hierarchical way, on two levels. The first level – L1 – is made of several instruction caches, each tightly coupled with, and private to, a CPU core. This first level has a relatively small capacity: 512B for each core 0 to 7. It supports a single outstanding request per core, and can be bypassed, which means that every request is then directly propagated to the second level of cache (default behavior after reset).

The second level of cache, called L1.5, is shared amongst the cores and has a 4kB capacity. It is in charge of dealing with cache misses from the L1 caches of all cores. It is a 4-way associative cache, and implements dual-banking to better spread L1.5 accesses from the different CPU cores, each bank dealing independently with its own refill requests. The replacement policy is pseudo-random: in case all ways are in use, the cache invalidates one random way using a code generated with a LFSR. L1.5 supports multiple outstatnding refill requests. This cache can also be disabled, which means that every request is then directly propagated toward L2 (default behavior after reset).

Besides, there is a special 2kB private L1 cache for master core 8, to deal with more complex local tasks. The ENABLE_SPECIAL_CORE_CACHE register is used to select 2 different modes for this cache: either used with shared L1.5 like other private caches, or used independently from L1.5 cache, in order not to interfere with the caching of the other cores.

All L1 caches have an optional feature to automatically prefetch the next 128-bit cache line – see ENABLE_L1_L15_PREFETCH register.

Note that the different service operations accessible through configuration register access (e.g. enabling, disabling, flushing, ...) are blocking, in the sense that the register access will be completed only at the end of the operation.

5.4.4.2 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
ENABLE	0x0	32	Cluster instruction cache enablement
FLUSH	0x4	32	Cluster instruction cache flush
L1_FLUSH	0x8	32	L1 cache flush
SEL_FLUSH	0xC	32	Selective instruction flush
ENABLE_SPECIAL_-CORE_CACHE	0x18	32	Core 8 cache sharing configuration to share
ENABLE_L1_L15_-PREFETCH	0x1C	32	L1 prefetch enablement

ENABLE

Cluster instruction cache enablement

Bit #	R/W	Name	Reset	Description
8:0	W	EN_PRI	0x0	Enable L1 private instruction cache (bit i configures L1 of core i): b0: disabled; b1: enabled
10:9	W	EN_SH	0x0	Enable banks of shared L1.5 cache (bit 9+i configures bank i): b0: disabled; b1: enabled

FLUSH

Cluster instruction cache flush

Bit #	R/W	Name	Reset	Description
8:0	W	FL_PRI	0x0	Write 1 to bit i to fully flush L1 private instruction cache of core i
10:9	W	FL_SH	0x0	Write 1 to bit 9+i to fully flush bank i of the shared L1.5 cache

L1_FLUSH

L1 cache flush

Bit #	R/W	Name	Reset	Description
8:0	W	L1_FL	0x0	Write 1 to bit i to fully flush L1 private instruction cache for core i

SEL_FLUSH

Selective instruction flush

Bit #	R/W	Name	Reset	Description
31:0	W	ADDR	0x0	Write an address to selectively flush it

ENABLE_SPECIAL_CORE_CACHE

Core 8 cache sharing configuration to share

Bit #	R/W	Name	Reset	Description
0	R/W	ENABLE	0x0	If b0, core 8 shares L1.5 with other cores; if b1, core 8 cache is independent from the other cores

ENABLE_L1_L15_PREFETCH

L1 prefetch enablement

Bit #	R/W	Name	Reset	Description
8:0	R/W	ENABLE	0x0	Enable the prefetchers for L1 private caches (bit i configures L1 of core i): b0: disabled; b1: enabled

5.4.5 Cluster DMA

Cluster DMA component manages data transfers between L2 and L1/TCDM memories. It offers:

- Two RX/TX full-duplex channels
- Up to 16 outstanding transfers between TCDM and L2 memories
- Linear or 2D transfer modes on both TCDM or L2 sides

DMA transfers are triggered through read and write operations to the CMD register. For a transaction with an ID, the first operation is to read the CMD register, which triggers a GET_TID command to retrieve the ID. This step is not done for a transaction with no ID. Then the characteristics of the transfer to be performed are configured by writing a NEW_TRANS command, a TCDM_ADDR command and a L2_ADDR command. In case of a 2D transfer on the L2 side, the commands L2_COUNT and L2_STRIDE are then sent. This step is not done if the transfer is linear on the L2 side. In case of a 2D transfer on the TCDM side, the commands TCDM_COUNT and TCDM_STRIDE are then sent. This step is not done if the transfer is linear on the TCDM side. At this stage the transfer is completely described, and a new transfer request can be programmed as needed.

5.4.5.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
CMD	0x0	32	Cluster DMA configuration register
STATUS	0x4	32	Cluster DMA status register

CMD

Cluster DMA configuration register

Bit #	R/W	Name	Reset	Description
31:0	R/W	COMMAND	0x00000000	DMA command – Format is command-dependent

STATUS

Cluster DMA status register

Bit #	R/W	Name	Reset	Description
15:0	R/W	TID_TR	0x0000	When read, bit i=1 means that the transfer with TID i is active. Write 1 to bit i to cancel/free the transfer with TID i.
31:16	R	TID_ALLOC	0x0000	Bit 16+i is read as 1 when transfer allocator with TID i is reserved, or 0 when it is free.

5.4.5.2 DMA commands encoding

Command name	Width	Command code	Description
GET_TID	32	N/A	(Triggered by reading CMD register before configuring a transfer) Get transfer identifier.
NEW_TRANS	32	N/A	Start to configure a new transfer.
TCDM_ADDR	32	N/A	Configure TCDM address of the transfer.
L2_ADDR	32	N/A	Configure L2 address of the transfer.
L2_COUNT	32	N/A	For a 2D transfer on L2 side, configure the length of the linear chucks of data for the transfer.
L2_STRIDE	32	N/A	For a 2D transfer on L2 side, configure the stride of the transfer.
TCDM_COUNT	32	N/A	For a 2D transfer on TCDM side, configure the length of the linear chucks of data for the transfer.
TCDM_STRIDE	32	N/A	For a 2D transfer on TCDM side, configure the stride of the transfer.

GET_TID

Bit #	Name	Description
3:0	TID	Value of the transfer identifier.

NEW_TRANS

Bit #	Name	Description
16:0	LEN	Transfer length in bytes.
17	TYPE	Transfer direction: b0: TCDM to L2; b1: L2 to TCDM.
18	INC	Set to 1 to configure an incremental transfer.
19	L2_2D	Transfer type on L2 side: b0: linear; b1: 2D transfer.
20	ELE	Set to 1 to enable event generation for the transfer.
21	ILE	Set to 1 to enable interrupt generation for the transfer.
22	BLE	Set to 1 to broadcast event and interrupts to all cluster cores. If 0, events and interrupts are only sent to the core which initiated the transfer.
23	TCDM_2D	Transfer type on TCDM side: b0: linear; b1: 2D transfer.

TCDM_ADDR

Bit #	Name	Description
31:0	ADDR	TCDM base address for the transfer.

L2_ADDR

Bit #	Name	Description
31:0	ADDR	L2 base address for the transfer.

L2_COUNT

Bit #	Name	Description
31:0	2D_CNT	Length of a linear part of a 2D transfer.

L2_STRIDE

Bit #	Name	Description
31:0	2D_STRIDE	Length of a stride of a 2D transfer (i.e. from a linear chunck of data to the next).

TCDM_COUNT

Bit #	Name	Description
31:0	2D_CNT	Length of a linear part of a 2D transfer.

TCDM_STRIDE

Bit #	Name	Description
31:0	2D_STRIDE	Length of a stride of a 2D transfer (i.e. from a linear chunck of data to the next).

5.4.6 Compressor/Decompressor

The Compressor/Decompressor block is used to retrieve compressed data from the L2 and uncompress them into the local memory (L1, or TCDM for tightly coupled data memory), or to compress data stored in the local memory and send them to L2. During a transfer from L2 to TCDM the data is decompressed using one of the 3 available schemes (T1, T2 and T3). During a transfer from TCDM to L2 the data can be compressed back using T1 compression scheme only.

Decompression Scheme	Description
T1	The decompression scheme *T1* consists in data expansion (signed/unsigned) to 8, 16 or 32 bits, within linear or 2D transfers from L2 to L1. Compressed data can be any size ranging from 1 to 31 bits.
T2	The decompression scheme *T2* consists in data transformation and expansion (signed/unsigned) to 8, 16 or 32 bits. Only linear transfers from L2 to L1 are supported. The transformation is performed via a look-up table (LUT), where the compressed data is interpreted as the LUT address, and the content of the LUT is the corresponding decompressed data. Compressed data (LUT addresses) size is ranging from 1 to 8 bits (i.e. maximum of 256 LUT entries).
T3	The decompression scheme *T3* consists in data transformation and expansion (signed/unsigned) to 8, 16 or 32 bits. Only linear transfers from L2 to L1 are supported. The *T3* scheme introduces a special symbol in the compressed data, which is encoded with a single bit (0). The T3 engine starts decompressing data and discriminates between occurancies of the special symbol (encoded with 0) and other data, which are compressed data using the LUT like in *T2*. Non special symbols are therefore encoded with a bit set to 1 (to discriminate from the special symbol) followed by the 1- to 8-bit LUT address.

Compression Scheme	Description
T1	The compression is performed by clipping 8-, 16- or 32-bit data from L1 to an arbitrary size ranging from 1 to 31 bits. Only linear transfers are supported.

5.4.6.1 Register map

Overview

Refer to GAP9 address map for the base address to be used.

Name	Offset	Width	Description
TCDM_ADDR_REG	0x0	32	Start address in TCDM (L1)
L2_ADDR_REG	0x4	32	Start address in L2
CONF_REG	0x8	32	Setup of the different parameters to configure compression/decompression
STATUS_REG	0xC	32	Busy status for decompressor engines T1, T2 and T3, and compressor engine T1
LUT_WRITE_REG	0x10	32	Write LUT content (for T2 and T3)
SPECIAL_SYMBOL_REG	0x14	32	For T3 decompression only, decompressed value of the special symbol compressed into a single '0' bit in L2
BIT_READ_REG	0x18	32	Number of bits read during the decompression/compression
MODE_REG	0x1C	32	Transfer mode (linear or 2D) for source and destination
SOFT_RESET_REG	0x20	32	Reset of compression/decompression engines and FIFOs (configuration registers are preserved)
CLOCK_ENABLE_REG	0x24	32	Clock gating control: enable/disable the clock of the compressor/decompressor
PUSH_CMD_REG	0x28	32	Trigger TX or RX transaction according to CONF_REG configuration
L2_COUNT_REG	0x30	32	Number of items to decompress after each stride from L2 to TCDM (Used only for 2D decompression)
L2_STRIDE_REG	0x34	32	Number of items to jump for every L2_COUNT_REG in the L2 (Used only for 2D decompression)
TCDM_COUNT_REG	0x38	32	Number of items to decompress after each stride from TCDM to L2 (Used only for 2D decompression)
TCDM_STRIDE_REG	0x3C	32	Number of items to jump for every TCDM_COUNT_REG in the L2 (Used only for 2D decompression)

TCDM_ADDR_REG

Start address in TCDM (L1)

Bit #	R/W	Name	Reset	Description
31:0	R/W	TCDM_START_ADDR	0x00000000	Start address for TCDM transfers (TX or RX)

L2_ADDR_REG

Start address in L2

Bit #	R/W	Name	Reset	Description
31:0	R/W	L2_START_ADDR	0x00000000	Start address for L2 transfers (TX or RX)

CONF_REG

Setup of the different parameters to configure compression/decompression

Bit #	R/W	Name	Reset	Description
1:0	R/W	decompr_mode	0x0	Decompression Mode: b00: T1; b01: T2; b11:T3; b10: reserved.
3:2	R/W	extension_type	0x3	Extension Type: b00: 8-bit; b01: 16-bit; b11: 32-bit; b10: reserved.
8:4	R/W	item_bit_width	0x04	Size of compressed data item in L2. Allowed sizes are 1 to 31 for T1, 1 to 15 for T2 and T3. Warning: An illegal configuration will be handled like a 1-bit width configuration
9	R/W	sign_extension	0x0	Sign extension during decompression: 0: Unsigned; 1: Signed.
12:10	R/W	start_bit	0x0	Bit offset (in the byte) for L2 start address during decompression or compression.
13	R/W	decompr_direction	0x0	Transfer direction: 0: TX (decompression); 1: RX (compression T1).
15:14	R/W	start_byte	0x0	Byte offset (in a 32-bit word) in L2 start address during decompression or compression.
31:16	R/W	items_to_transfer	0x0000	Total items to compress/decompress during a transfer.

STATUS_REG

Busy status for decompressor engines T1, T2 and T3, and compressor engine T1

Bit #	R/W	Name	Reset	Description
0	R	T1_DECOMPR_BUSY	0x0	Status of the T1 decompression engine: if 1, engine is busy.
1	R	T2_DECOMPR_BUSY	0x0	Status of the T2 decompression engine: if 1, engine is busy.
2	R	T3_DECOMPR_BUSY	0x0	Status of the T3 decompression engine: if 1, engine is busy.
3	R	T1_COMPR_BUSY	0x0	Status of the T1 compression engine: if 1, engine is busy.

LUT_WRITE_REG

Write LUT content (for T2 and T3)

Bit #	R/W	Name	Reset	Description
15:0	W	LUT_DATA	0x0000	Data to write in the LUT
23:16	W	LUT_ADDR	0x00	Address of the LUT to be written

SPECIAL_SYMBOL_REG

For T3 decompression only, decompressed value of the special symbol compressed into a single '0' bit in L2

Bit #	R/W	Name	Reset	Description
31:0	R/W	SPECIAL_SYMBOL	0xABBAABBA	Special symbol compressed as a single '0' bit for T3 decompression

BIT_READ_REG

Number of bits read during the decompression/compression

Bit #	R/W	Name	Reset	Description
31:0	R	BIT_READ	0x00000000	Number of bits read at the end of decompression

MODE_REG

Transfer mode (linear or 2D) for source and destination

Bit #	R/W	Name	Reset	Description
1:0	R/W	TRANSF_MODE	0x0	Transfer mode: 0: linear L2/linear TCDM; 1: 2D L2/linear TCDM; 2: linear L2/2D TCDM; 3: reserved. Warning: Only T1 decompression supports a non-0 setting.

SOFT_RESET_REG

Reset of compression/decompression engines and FIFOs (configuration registers are preserved)

Bit #	R/W	Name	Reset	Description
0	W	SOFT_RESET	0x0	Writing any value generates a reset to clear all engines and FIFOs. Configuration registers are not reset.

CLOCK_ENABLE_REG

Clock gating control: enable/disable the clock of the compressor/decompressor

Bit #	R/W	Name	Reset	Description
0	W	CLOCK_ENABLE	0x0	Set to 1 to enable the clock for the compressor/decompressor block. Access to configuration registers remains active when the clock is disabled.

PUSH_CMD_REG

Trigger TX or RX transaction according to CONF_REG configuration

Bit #	R/W	Name	Reset	Description
0	W	TRIGGER	0x0	Write 1 to trigger a TX or RX transfer, according to configured settings.

L2_COUNT_REG

Number of items to decompress after each stride from L2 to TCDM (Used only for 2D decompression)

Bit #	R/W	Name	Reset	Description
15:0	R/W	L2_LINEAR_COUNT	0x0000	For L2 2D transfers, number of items for each linear part of the transfer, minus 1 (e.g. set to 7 for a 8-item linear part).

L2_STRIDE_REG

Number of items to jump for every L2_COUNT_REG in the L2 (Used only for 2D decompression)

Bit #	R/W	Name	Reset	Description
15:0	R/W	L2_STRIDE_COUNT	0x0000	For L2 2D transfers, number of items in a stride (i.e. from a linear chunck of data to the next), minus 1 (e.g if the first item of the next linear chunk of data is 16 items after the first item of current chunk, then set to 15).

TCDM_COUNT_REG

Number of items to decompress after each stride from TCDM to L2 (Used only for 2D decompression)

Bit #	R/W	Name	Reset	Description
15:0	R/W	TCDM_LINEAR_-COUNT	0x0000	For TCDM 2D transfers, number of items for each linear part of the transfer, minus 1.

TCDM_STRIDE_REG

Number of items to jump for every TCDM_COUNT_REG in the L2 (Used only for 2D decompression)

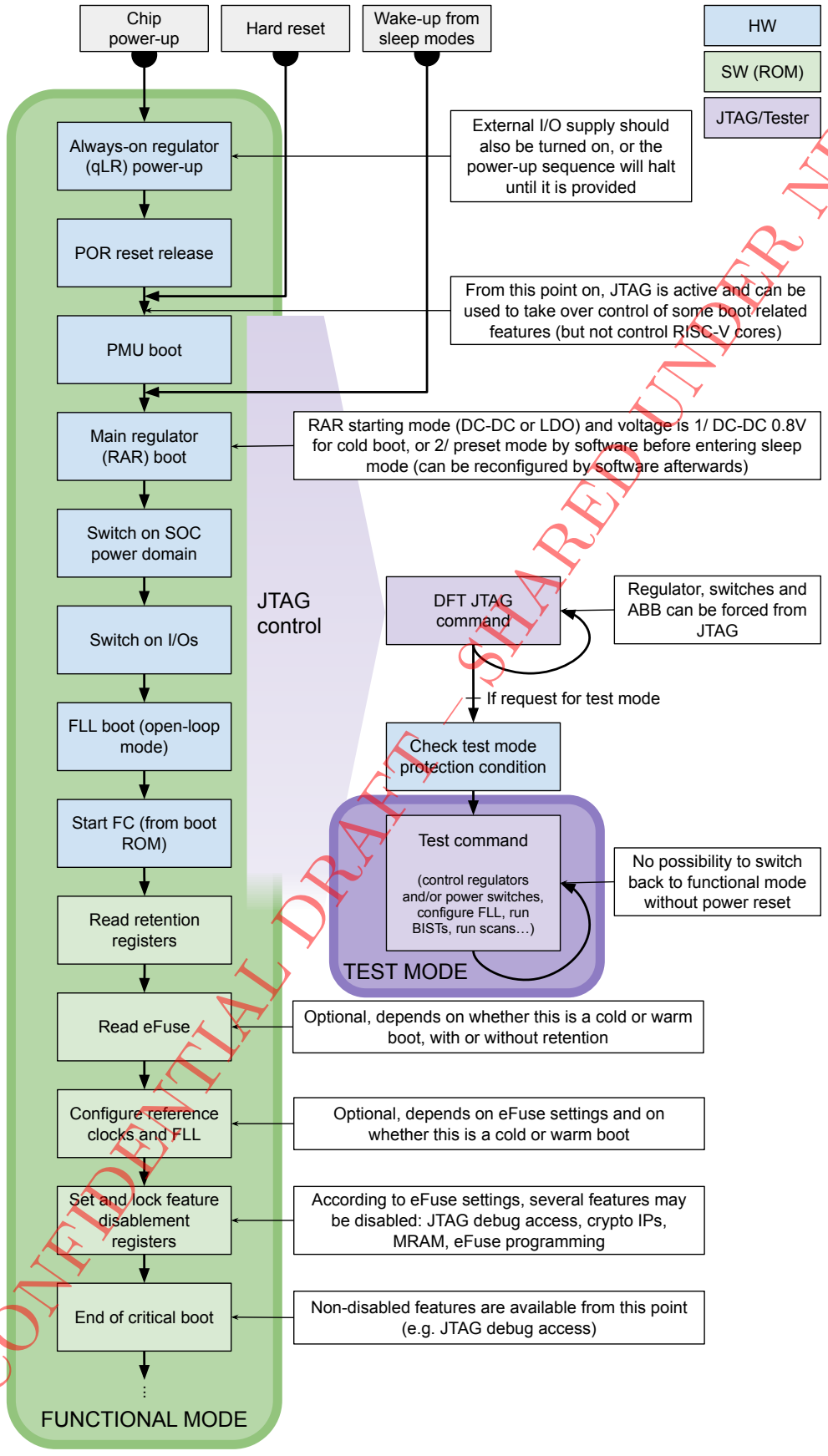
Bit #	R/W	Name	Reset	Description
15:0	R/W	TCDM_STRIDE_-COUNT	0x0000	For TCDM 2D transfers, number of items in a stride (i.e. from a linear chunck of data to the next), minus 1.

Chapter 6

Boot Process

The next figure describes the first steps of GAP9 boot process, as executed by hardware then through the execution of boot ROM software. As shown in the figure, the same diagram, with different entry points, applies to power-on boot, hard reset and wake-up from sleep modes. Always-on (retentive) registers are then used to discriminate the mode that the chip should eventually enter.

Those steps are further detailed in the following sections. The Local Supply Monitoring section also gives details on how power supply monitoring is involved in the initial steps of the boot process, at chip power-up.



Note: On top of discriminating cold boot from a warm reset or a wake up from retentive or non-retentive sleep, the runtime software can use always-on registers to configure, before entering sleep, the power mode the chip

should be turned into.

6.1 Power-Up

The first steps of the boot process relate to power supply, and differ for cold boot and wake-up from sleep modes.

6.1.1 Cold Boot

In the case of a cold boot, GAP9 first requires its external `VBAT` power supply to start its internal low power LDO regulator (the qLR), used to power always-on logic.

Once the internal power supply is stable, the internal reset is released by the POR (Power-On Reset), enabling the power management unit (PMU) to execute its startup sequence. This sequence is similar to the Seq. 1 "Wake up SOC" shown in the table of the Transitions Between Power Modes section, and basically consists in enabling the main voltage regulator (the RAR), and switching on the I/Os and the SOC power domain. The RAR ouput voltage is set to 0.8V at startup.

Note: If the `VDDIO` supply is not present when power-on reset is released, the PMU will halt its start-up sequence, to prevent GAP9 to boot without its I/Os being properly powered. It is recommended to maintain GAP9 reset input low until `VDDIO` is powered and stabilized.

At this stage, GAP9 is in the NOMINAL_VOLTAGE power mode (see Predefined Configurations). The FLL then starts in open-loop mode (i.e. without using the reference clock), at a default frequency of approximately 50MHz. This allows the *Fabric Controller* core to start executing instructions from the boot ROM.

6.1.2 Boot From Sleep Mode

In sleep mode, GAP9 LDO remains on, allowing the PMU to stay in a low power standby mode. Depending on the selected wake-up configuration (see section Wake-Up from Sleep Modes for more details), one of the predefined GAP9 wake-up sequences is triggered to power-up the SOC power domain and optionally other domains depending on the selected sequence. As an example, selecting sequence 3 allows to automatically enable SOC, I/Os, cluster and MRAM at wake-up time, without the need for the software to trigger additional sequences.

It is also possible to select a specific wake-up configuration for the RAR regulator, by preconfiguring it when going to sleep. This configuration will be automatically used when re-enabling the RAR during wake-up.

Like for cold boot, the FLL then starts, either in open-loop mode or using its previous configuration in case of retentive sleep, and the FC core starts its execution.

6.2 FC Boot From ROM

Once powered and clocked, the FC core starts executing the boot ROM software. The first step is to read the always-on registers, which are configured by GAP9 runtime before entering sleep mode. Their value allows the ROM software to discriminate the type of boot it is dealing with: cold boot, wake-up from deep sleep, wake-up from retentive sleep.

For a wake-up from retentive sleep, the execution directly jumps back to interrupted application software. Otherwise, the boot software accesses the eFuses, which determine a number of parameters and options to be used to complete the boot process: how to configure the clocks, from which peripheral to boot, is the firmware encrypted, etc. The GAP9 ROM section details the different parameters stored in the eFuse and their impact on the boot process.

Note: Being eFuses and not regular registers, read and write accesses are done using a specific interface (see eFuse interface description). Any bit set to 1 cannot be set back to 0. This also ensures that it is not possible to revert a feature disablement configuration.

Once the eFuse dependent parameters and configurations taken into account, the ROM leaves the critical boot section, and proceeds with the boot from the selected source.

6.3 JTAG and Boot Process

During the boot process, GAP9 features that may be controlled by JTAG are limited, in order to prevent security breaches which could interfere with sensitive operations such as proper disablement of features forbidden by eFuse configuration, access to key management, or turning the chip to test mode. In particular it is not possible to take control on a CPU through JTAG before the end of the critical boot section. For the same reason, hardwired hardware features prevent from switching back to functional mode after entering test mode without a complete reboot, i.e. switching off then on the external chip power supply.

Some features can be forced or probed through JTAG during the boot process, as a mean to debug system start-up. Please contact Greenwaves Technologies if such an access is needed. Features which can be force or monitored through JTAG include:

- RAR voltage regulator,
- Power switches (power domains and I/Os),
- Automatic Back Bias generator (ABB).

Chapter 7

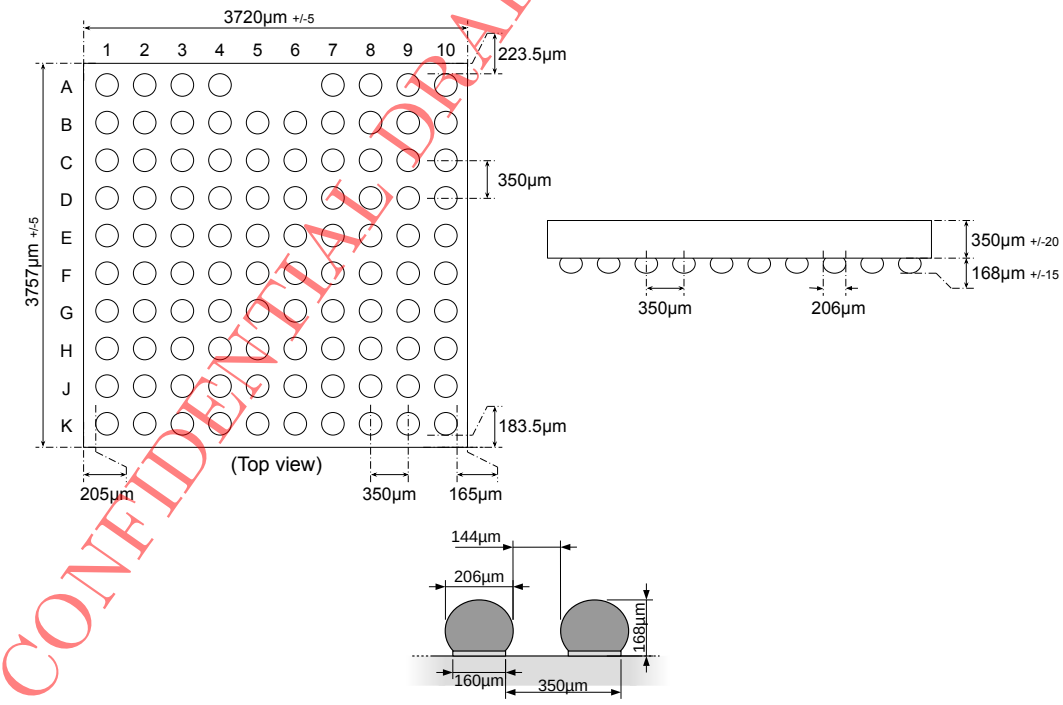
WLCSP Package

GAP9 is available as a WLCSP (Wafer-Level Chip Sized Package) package, which offers a reduced form factor of 3.72mm × 3.76mm. Its balls are arranged in a 10×10, .35mm pitch array. Due to the limited physical size of WLCSP, its number of balls do not allow to output every I/Os of GAP9 and not all functional pins are available: single memory interface, no CPI interface, single lane for CSI-2 interface, etc.

A BGA (Ball-Grid Array) package is planned in the future. Its size will be 5.5mm × 5.5mm, and its balls will be arranged in a 12×12, .4mm pitch array. The BGA package will give access to every I/Os of GAP9, i.e. all features will be usable.

7.1 Mechanical Information

The following figures shows the physical dimensions of the WLCSP.



7.2 Pin Description

Refer to section SoC Controller for a description of the configuration registers related to I/Os.

Pad ID	Ball	I/O Function				Always-on / Switched[3]	Drive Strength[5]	Schmitt Trigger[6]	Optional Pull-Up[1]	Optional Pull-Down[1]	Wake-up Source[8]
		Alternate 0[11]	Alternate 1[2]	Alternate 2	Alternate 3						
38	A1	SPI1_SDO	GPIO38	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
89	A2	WAKEUP_-SPI2_CS0	GPIO89	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	A3	VSS				N/A (power supply)					
–	A4	CSI2_VCCA065				N/A (power supply)					
–	(A5)	(Ball Does Not Exist)				N/A					
–	(A6)	(Ball Does Not Exist)				N/A					
–	A7	VSS (RAR_AVS)				N/A (power supply)					
–	A8	VBAT				N/A (power supply)					
–	A9	VSAFE				N/A					
–	A10	VREG_OPM				N/A					
40	B1	I2C0_-SDA[11]	GPIO40	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
33	B2	SPI1_SCK	GPIO33	USART3_-CLK	–	Always On	2/4/8/12mA	Option	Y	Y	
88	B3	WAKEUP_-SPI2_SDO	GPIO88	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	B4	VDD_LOGIC				N/A (power supply)					
–	B5	CSI2_VSS				N/A (power supply)					
–	B6	CSI2_-DATAP0	–	–	–	[4]	N/A	N/A	N	N	
–	B7	RAR_LX				N/A					
–	B8	RAR_GNDSENSE				N/A					
–	B9	VSS				N/A (power supply)					
–	B10	VDDIO				N/A (power supply)					
43	C1	I2C1_SCL[11]	GPIO43	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
35	C2	SPI1_CS1[11]	GPIO35	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	C3	SLOW_-XTAL_XA	–	–	–	Always On	N/A	N/A	N	N	
–	C4	SLOW_-XTAL_XB	–	–	–	Always On	N/A	N/A	N	N	

Pad ID	Ball	I/O Function				Always-on / Switched[3]	Drive Strength[5]	Schmitt Trigger[6]	Optional Pull-Up[1]	Optional Pull-Down[1]	Wake-up Source[8]
		Alternate 0[11]	Alternate 1[2]	Alternate 2	Alternate 3						
–	C5	VDDIO				N/A (power supply)					
–	C6	CSI2_-DATAN0	–	–	–	[4]	N/A	N/A	N	N	
–	C7	CSI2_VCCA18				N/A (power supply)					
–	C8	RAR_VSENSE				N/A					
–	C9	VQPS_FUSE_1V8				N/A (power supply)					
–	C10	VDD_LOGIC				N/A (power supply)					
84	D1	JTAG_-TMS[2]	GPIO84	USART4_-CTS	–	Always On	2/4/8/12mA	Option	Y	Y	
41	D2	I2C0_SCL[11]	GPIO41	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
68	D3	PWM1[11]	GPIO68	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	D4	RESETN	–	–	–	Always On	N/A	N/A	N	N	
87	D5	WAKEUP_-SPI2_SDI	GPIO87[10]	–	–	Always On	2/4/8/12mA	Option	Y	Y	
86	D6	WAKEUP_-SPI2_SCK	GPIO86[10]	–	–	Always On	2/4/8/12mA	Option	Y	Y	
–	D7	CSI2_CLKP	–	–	–	[4]	N/A	N/A	N	N	
–	D8	QLR_GNDSENSE				N/A					
10	D9	HYPER0_-OSPI_CSN0	GPIO10	–	–	Switched	1/2/4/8mA	Y	Y	Y	
9	D10	HYPER0_-OSPI_DQ7	GPIO9	–	–	Switched	1/2/4/8mA	Y	Y	Y	
–	E1	FAST_-XTAL_XA[7]	–	–	–	Always On	N/A	N/A	N	N	
82	E2	JTAG_-TDI[2]	GPIO82	USART4_RX	–	Always On	2/4/8/12mA	Option	Y	Y	
39	E3	SPI1_SDI	GPIO39	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
34	E4	SPI1_CS0[11]	GPIO34	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
67	E5	PWM0[11]	GPIO67	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	E6	*Not Connected*				N/A					
–	E7	CSI2_CLKN	–	–	–	[4]	N/A	N/A	N	N	
11	E8	HYPER0_-OSPI_CSN1	GPIO11	–	–	Switched	1/2/4/8mA	Y	Y	Y	
7	E9	HYPER0_-OSPI_DQ5	GPIO7	–	–	Switched	1/2/4/8mA	Y	Y	Y	

Pad ID	Ball	I/O Function				Always-on / Switched[3]	Drive Strength[5]	Schmitt Trigger[6]	Optional Pull-Up[1]	Optional Pull-Down[1]	Wake-up Source[8]
		Alternate 0[11]	Alternate 1[2]	Alternate 2	Alternate 3						
6	E10	HYPER0_-OSPI_DQ4	GPIO6	–	–	Switched	1/2/4/8mA	Y	Y	Y	
–	F1	FAST_-XTAL_XB[7]	–	–	–	Always On	N/A	N/A	N	N	
85	F2	JTAG_-NTRST[2]	GPIO85	USART4_-RTS	–	Always On	2/4/8/12mA	Option	Y[1]	Y[1]	
83	F3	JTAG_-TDO[2]	GPIO83	USART4_TX	–	Always On	2/4/8/12mA	Option	Y	Y	
42	F4	I2C1_-SDA[11]	GPIO42	–	–	Always On	2/4/8/12mA	Option	Y	Y	Y
–	F5	VDDIO				N/A (power supply)					
–	F6	*Not Connected*				N/A					
–	F7	*Not Connected*				N/A					
8	F8	HYPER0_-OSPI_DQ6	GPIO8	–	–	Switched	1/2/4/8mA	Y	Y	Y	
–	F9	VDDIO				N/A (power supply)					
–	F10	VSS				N/A (power supply)					
–	G1	VDD_LOGIC				N/A (power supply)					
–	G2	VDDIO				N/A (power supply)					
–	G3	VSS				N/A (power supply)					
81	G4	JTAG_-TCK[2]	GPIO81	USART4_-CLK	–	Always On	2/4/8/12mA	Option	Y	Y	
45	G5	I2C2_SCL[11]	GPIO45	–	–	Switched	2/4/8/12mA	Option	Y	Y	
48	G6	SAI0_SCK	GPIO48	USART2_-CLK	–	Switched	2/4/8/12mA	Option	Y	Y	
49	G7	SAI0_WS	GPIO49	–	–	Switched	2/4/8/12mA	Option	Y	Y	
–	G8	*Not Connected*				N/A					
–	G9	*Not Connected*				N/A					
–	G10	VDD_LOGIC				N/A (power supply)					
56	H1	SAI2_SCK	GPIO56	SPI2_SCK	–	Switched	2/4/8/12mA	Option	Y	Y	
57	H2	SAI2_WS	GPIO57	SPI2_CS0	–	Switched	2/4/8/12mA	Option	Y	Y	
58	H3	SAI2_SDI	GPIO58	SPI2_SDI	–	Switched	2/4/8/12mA	Option	Y	Y	
59	H4	SAI2_SDO	GPIO59	SPI2_SDO	–	Switched	2/4/8/12mA	Option	Y	Y	
46	H5	I3C_SDA	GPIO46	I2C3_SDA	SPI0_SDIO2	Switched	2/4/8/12mA	Option	Y	Y	
51	H6	SAI0_SDO	GPIO51	–	–	Switched	2/4/8/12mA	Option	Y	Y	

Pad ID	Ball	I/O Function				Always-on / Switched[3]	Drive Strength[5]	Schmitt Trigger[6]	Optional Pull-Up[1]	Optional Pull-Down[1]	Wake-up Source[8]
		Alternate 0[11]	Alternate 1[2]	Alternate 2	Alternate 3						
53	H7	SAI1_WS	GPIO53	SPI2_CS1	–	Switched	2/4/8/12mA	Option	Y	Y	
3	H8	HYPER0_-OSPI_DQ1	GPIO3	–	–	Switched	1/2/4/8mA	Y	Y	Y	
4	H9	HYPER0_-OSPI_DQ2	GPIO4	–	–	Switched	1/2/4/8mA	Y	Y	Y	
5	H10	HYPER0_-OSPI_DQ3	GPIO5	–	–	Switched	1/2/4/8mA	Y	Y	Y	
62	J1	USART0_-CTS[11]	GPIO62	–	–	Switched	2/4/8/12mA	Option	Y	Y	
63	J2	USART0_-RTS[11]	GPIO63	–	–	Switched	2/4/8/12mA	Option	Y	Y	
61	J3	USART0_-TX[11]	GPIO61	–	–	Switched	2/4/8/12mA	Option	Y	Y	
60	J4	USART0_-RX[11]	GPIO60	–	–	Switched	2/4/8/12mA	Option	Y	Y	
47	J5	I3C_SCL	GPIO47	I2C3_SCL	SPI0_SDIO3	Switched	2/4/8/12mA	Option	Y	Y	
52	J6	SAI1_SCK	GPIO52	–	–	Switched	2/4/8/12mA	Option	Y	Y	
–	J7	VDDIO				N/A (power supply)					
–	J8	VSS				N/A (power supply)					
12	J9	HYPER0_-OSPI_DS	GPIO12	–	–	Switched	1/2/4/8mA	Y	Y	Y	
2	J10	HYPER0_-OSPI_DQ0	GPIO2	–	–	Switched	1/2/4/8mA	Y	Y	Y	
64	K1	USART0_-CLK/FAST_-REF_CK[9]	GPIO64[9]	–	–	Switched	2/4/8/12mA	Option	Y	Y	
–	K2	VDD_LOGIC				N/A (power supply)					
–	K3	VDDIO				N/A (power supply)					
44	K4	I2C2_-SDA[11]	GPIO44	–	–	Switched	2/4/8/12mA	Option	Y	Y	
50	K5	SAI0_SDI	GPIO50	–	–	Switched	2/4/8/12mA	Option	Y	Y	
54	K6	SAI1_SDI	GPIO54	SPI2_CS2	SPI2_SDIO2	Switched	2/4/8/12mA	Option	Y	Y	
55	K7	SAI1_SDO	GPIO55	SPI2_CS3	SPI2_SDIO3	Switched	2/4/8/12mA	Option	Y	Y	
–	K8	VDD_LOGIC				N/A (power supply)					

Pad ID	Ball	I/O Function				Always-on / Switched[3]	Drive Strength[5]	Schmitt Trigger[6]	Optional Pull-Up[1]	Optional Pull-Down[1]	Wake-up Source[8]
		Alternate 0[11]	Alternate 1[2]	Alternate 2	Alternate 3						
1	K9	HYPER0_-OSPI_CK	GPIO1	–	–	Switched	1/2/4/8mA	Y	Y	Y	
0	K10	HYPER0_-OSPI_CKN	GPIO0	–	–	Switched	1/2/4/8mA	Y	Y	Y	

Notes:

[1] Most I/Os have optional pull capability, either pull-up (PU) or pull-down (PD). Equivalent pull resistor is in the 50-150Kohm range (except if specified otherwise). When present, the pull resistor is disabled at power-up and hardware reset, except for JTAG_NTRST (F2) which is always weakly pulled high at reset.

[2] All pins that have alternate functionalities start up at reset in Alt. 1 mode (that is, GPIO), configured as floating input (i.e. without pull-up/down), with the exception of the JTAG interface, which is active at reset. Note: The primary (ROM'd) boot code will then configure the direction of some I/Os it needs to later fetch secondary boot code from an external interface.

[3] Internal switches cut off power supplies of a number of functional I/Os when not in use, for example when GAP9 is in Sleep Mode. Those I/Os are marked as 'Switched' in the above table. Other I/Os stay always on: these are marked as such in the table. They can be used for example to set external hardware during sleep. The structure of *Switched* I/Os is such that they will be stuck at logic '0', i.e. VSS, when switched off. Therefore, **those I/Os should not be driven high** by an external component or pull-up when they are off.

[4] CSI-2 interface can be selectively switched off by powering off its dedicated analog power supplies.

[5] Output drive of digital I/Os is programmable. Default value is the lowest listed value.

[6] Schmitt Trigger is present on some pads and optional on some others, as indicated. Optional means it is not activated by default but can be enabled under software control.

[7] If the 'fast' oscillator is fed directly with an external clock, its swing must be restricted to 0-0.8V. In addition, during sleep modes this oscillator may be either kept on or switched off, under software control.

[8] Balls flagged in the "wake-up source" column can be configure under software control in order to wake up GAP9 from sleep when a specific logic level is applied onto the chip ball (see SLEEP_GPIO_CTRL and SLEEP_CTRL registers).

[9] This pin can be used as external reference clock (0-50MHz, 0-1.8V swing) under control of configuration bits, in which case the "fast" crystal oscillator is not used and crystal is not required. **Note that the pad must then be configured in GPIO input mode.** Pay also attention to Note (3) above.

[10] At power-up/reset, GPIO86 and GPIO87 can be used as *bootsel* pads to determine the boot source (see GAP9 ROM section for more details).

[11] Pin function Alt. 0 of designated I/Os can be replaced under software control by any signal from the following list (also see this note on configuring reprogrammable pads for more details on use and limitations):

I2C0_SDA	PWM0	UART0_RX	UART2_RX	SPI0_CS0
I2C0_SCL	PWM1	UART0_TX	UART2_TX	SPI0_CS1
I2C1_SDA	PWM2	UART0_CTS	UART2_CTS	SPI0_CS2
I2C1_SCL	PWM3	UART0_RTS	UART2_RTS	SPI0_CS3
I2C2_SDA	PWM4	UART1_RX	UART3_TX	SPI1_CS0
I2C2_SCL	PWM5	UART1_TX	UART3_RX	SPI1_CS1
CSI2_CCI_SDA	PWM6	UART1_CTS		
CSI2_CCI_SCL	PWM7	UART1_RTS		

Chapter 8

Operating Conditions

8.1 Voltages

Name	Description	Minimum	Typical	Maximum
VBAT	Main supply for GAP9 internal logic	1.9V 1.71V[1]	– 1.8V	5.5V
VDDIO	Power supply for I/Os and eMRAM	1.62V	1.8V	1.98V
CSI_VCCA18[2]	Analog supply for CSI-2 interface	1.62V	1.8V	1.98V
CSI_-VCCA065[2]	Analog supply for CSI-2 interface	0.585V	0.65V-0.8V	0.98V
VQPS_-FUSE_1V8[3]	Power supply for eFuse programming	1.71V	1.8V	1.89V
VIL_{AON}	Low-level input voltage for always-on digital I/Os			0.35×VDDIO
VIH_{AON}	High-level input voltage for always-on digital I/Os	0.65×VDDIO		
VOL_{AON}	Low-level output voltage for always-on digital I/Os			0.4V
VOH_{AON}	High-level output voltage for always-on digital I/Os	VDDIO - 0.4V		
VIL_{SWITCH}	Low-level input voltage for switchable digital I/Os			0.3×VDDIO
VIH_{SWITCH}	High-level input voltage for switchable digital I/Os	0.7×VDDIO		
VOL_{SWITCH}	Low-level output voltage for switchable digital I/Os			0.45V
VOH_{SWITCH}	High-level output voltage for switchable digital I/Os	VDDIO - 0.45V		VDDIO
V_{PAD}	Voltage at digital I/O	-0.3V		VDDIO + 0.3V

[1] VBAT supply below 1.9V is only supported for temperatures above -25°C.
[2] CSI-2 analog supplies can be switched off if this interface is not used.
[3] When not programming eFuse, VQPS_FUSE_1V8 can be grounded, left floating or powered in the 1.62V-1.98V range. Maximum current drawn on this supply when programming eFuses is 22mA – typical current drawn is 12mA.

8.2 Junction Temperature

Operating mode	Description	Minimum	Maximum
Normal	Standard operating mode when VBAT $\geq$ 1.9V	-40°C	125°C
Low VBAT	Standard operating mode when VBAT $<$ 1.9V	-25°C	125°C
EFuse programming	Reduced temperature range for eFuse programming	0°C	100°C

8.3 Frequencies

[Assessment of maximum frequencies for 0.7V and 0.75V are ongoing.]

8.3.1 Externally-Provided Clocks

The table below gives the maximum frequency of every clock signal provided externally to GAP9.

Clock	Description	Max frequency
REF FAST clock[4]	Fast reference clock	50MHz
REF SLOW clock[5]	Slow reference clock	32.768kHz
JTAG clock	JTAG clock	12.5MHz
SAIi external clock	SAI interface i clock when externally provided	50MHz@0.8V 40MHz@0.65V
CSI-2 input clock	CSI-2 differential clock from external camera (used in DDR mode)	750MHz
CPI clock	CPI clock from external camera	100MHz
SPI0 slave clock	SPI interface 0 clock in slave mode, coming from the pad	53MHz@0.8V 48MHz@0.65V
SPI1 & SPI2 slave clocks	SPI interfaces 1 and 2 clocks in slave mode, coming from the pad	20MHz
SPI3 slave clock	SPI interface 3 clock in slave mode, coming from the pad	46MHz@0.8V 40MHz@0.65V

8.3.2 Internally Generated Clocks

Note: Most of I/O interfaces use a configurable clock to control their internal logic. The real frequency at the I/O interface may also depend on each interface controller internal behavior, and on the used register configuration. Please refer to the corresponding section of this datasheet for more details on the clocking scheme and options for a given peripheral.

The next table gives the maximum frequency of every internally generated clock. These clocks are used internally to the chip and/or for I/O peripherals' signals. For each clock, the reference clock used for its generation is also given.

[4] Minimum allowed frequency of REF FAST when using the oscillator is 1MHz.
[5] REF SLOW clock is actually internally generated by an on-chip oscillator, from an external quartz. It has a fixed frequency, to guarantee the precision of the real time clock.

Clock	Base clock	Description	Max frequency
DIV REF FAST clock	REF FAST clock	Reference clock with reduced frequency for events, timers, IDLE mode, etc.	12.5MHz
SOC clock	From FLL	Main clock for SOC domain	370MHz@0.8V 325MHz@0.75V **(TBC)** 280MHz@0.7V **(TBC)** 240MHz@0.65V
SFU clock	From FLL	Main clock for the SFU	370MHz@0.8V 325MHz@0.75V **(TBC)** 280MHz@0.7V **(TBC)** 240MHz@0.65V
CLUSTER clock	From FLL	Main clock for the cluster	370MHz@0.8V 325MHz@0.75V **(TBC)** 280MHz@0.7V **(TBC)** 240MHz@0.65V
PERIPH clock	From FLL	Main clock for the I/O peripherals	370MHz@0.8V 325MHz@0.75V **(TBC)** 280MHz@0.7V **(TBC)** 240MHz@0.65V
MEMORYi clock	PERIPH clock	Memory interface i clock (used in DDR mode)	185MHz@0.8V 120MHz@0.65V
MRAM clock	PERIPH clock	Embedded MRAM clock	37MHz@0.8V **(TBC)** 34MHz@0.65V **(TBC)**
SPI0 master clock	PERIPH clock	SPI interface 0 clock, in master mode	53MHz@0.8V 48MHz@0.65V
SPI1 & SPI2 master clocks	PERIPH clock	SPI interfaces 1 and 2 clocks, in master mode	20MHz
SPI3 master clock	PERIPH clock	SPI interface 3 clock, in master mode	46MHz@0.8V 40MHz@0.65V
SAIi internal clock	PERIPH clock	SAI interface i clock when internally provided	53MHz@0.8V 48MHz@0.65V
I3C clock[6]	PERIPH clock	I3C controller base clock	92MHz@0.8V 80MHz@0.65V
CCI clock[7]	PERIPH clock	Clock of the camera control interface of the CSI-2 controller	10MHz
CSI-2 APB clock	PERIPH clock	Clock of APB interface of the CSI-2 controller	60MHz
CSI-2 PIXEL clock	PERIPH clock	Clock used to retrieve pixel data at CSI-2 interface output	370MHz@0.8V 240MHz@0.65V

[6] The actual datarate used on the I3C interface is derived from this clock according to the configuration of the registers of the controller. It must comply with the frequency allowed by the standard (up to 12.5MHz in SDR mode, 25MHz in HDR-DDR mode).

[7] The actual datarate used on the CCI interface matches this clock divided by 2 – most cameras support FM (400kHz) or FM+ (1000kHz) as defined by I2C standard.

8.3.3 Maximum Digital I/O Signal Speed

The table below gives the maximum frequency achievable by the digital I/O signals in the chip, depending on the type of I/O, the setting of the drive (PAD_*_DRIVE_STRENGTH fields, see SoC Controller) and the external load capacitance. The real performance is highly dependent on the particular use of the chip in the system.

Note: The following numbers are the intrinsic limitation of the physical I/Os. For most use cases, the most limiting factor is the maximum frequency supported by the peripheral controllers, as given later in this section.

I/O type	Drive setting	Maximum frequency (MHz)	Load capacitance (pF)
Always on	b11	50	15
Switchable I/Os, excluding memory interface I/Os	b00	100	2
	b01		4
	b10		8
	b11		16
	b00	25	45
	b01	40	
	b10	70	
	b11	80	
Memory interfaces switchable I/Os (HYPER*_-OSPI_*)	b00	300	1
	b01		2
	b10		4
	b11		8

The maximum frequencies supported by the different I/O protocols are listed in the following table:

Interface	Maximum data frequency (MHz)		Note
	@0.65V	@0.8V	
JTAG	12.5	12.5	
HYPER0_OSPI (DDR)	240	370	DDR transfer
HYPER1_OPSI (DDR)	240	370	DDR transfer
SPI0 (master or slave)	48	53	
SPI1 (master or slave)	20	20	
SPI2 (master or slave)	20	20	
SPI3 (master or slave)	40	46	
U(S)ART0	Limited by the max frequency of the digital I/O used		
U(S)ART1	Limited by the max frequency of the digital I/O used		
U(S)ART2	Limited by the max frequency of the digital I/O used		
U(S)ART3	Limited by the max frequency of the digital I/O used		
U(S)ART4	Limited by the max frequency of the digital I/O used		
I2C0	1	1	Defined by the I2C standard[8] (FM+ communication)
I2C1	1	1	
I2C2	1	1	
I2C3	1	1	
SAI0 (external clock)	40	50	
SAI0 (internal clock)	48	53	
SAI1 (external clock)	40	50	
SAI1 (internal clock)	48	53	
SAI2 (external clock)	40	50	
SAI2 (internal clock)	48	53	
CSI-2 (DDR)	1500	1500	DDR transfer
CCI	1	1	Defined by the I2C standard[8]
CPI	100	100	
I3C (SDR mode)	12.5	12.5	Defined by the I3C standard[8]
I3C (HDR-DDR mode)	25	25	DDR transfer – Defined by the I3C standard[8]

8.4 MRAM Reliability

GAP9 embedded MRAM meets the following bit endurance and retention characteristics for all supported temperatures.

Parameter	Minimum specification	Junction temperature	Unit
Data endurance	100,000	-40°C – 125°C	cycles
Data retention	20	125°C	years

[8] This maximum frequency is defined by standard. GAP9 supports higher transfer rates if needed, through configuration.

8.5 Electrical Sensitivity

(TBC)

GAP9 is compliant with electrostatic discharge (ESD) charge device model (CDM) and human body model (HBM), and latch-up (LU) immunity standards as defined in the table below.

Rating	Standard	Class	Max value
ESD (HBM)	ESDA/JEDEC JS-001-2017	2	2000V
ESD (CDM)	ESDA/JEDEC JS002	C2a	500V
LU	JESD78	I.A	Vmax × 1.5; 100mA

8.6 Power Consumption

[To be completed]

This section provides the typical power consumption of GAP9 for different operating modes and applications. Note that the power consumption is highly dependent on operating conditions, such as activity, power supply and temperature. The impact of temperature is paticularly significant at low activity, where leakage is dominent. Unless otherwise specified, those results have been measured on a typical process corner GAP9 chip, at ambiant temperature, with a 1.8V VBAT power supply.

8.6.1 Sleep Modes

Number of retentive banks		Power consumption (µW)		
Private L2	Interleaved L2	Deep sleep	Retentive sleep	Light sleep
0	0	41	–	–
1	0	–	266	275
2	0	–	277	281
2	4	–	338	310
2	8	–	400	335
2	12	–	468	360

8.6.2 Baseline Power Consumption

These measures show the consumption of the GAP9 chip with very low activity, on different hypotheses of clock and L2 memory configurations. Common settings are:

- VDD_LOGIC is set to 0.65V.
- SOC power domain is on, whereas cluster, SFU, MRAM and CSI-2 power domains are off.
- Switchable I/Os are on, but I/O interfaces are inactive (gated), apart from the JTAG interface.
- Active FLL DCOs are configured at 10MHz.
- FC core is on, running an idle loop.

REF FAST clock oscillator	FLL configuration	L2 memory	VBAT power (mW)	VDD_LOGIC power (mW)	VDDIO power (mW)
Active	4 DCOs, closed-loop	All banks on	2.59	1.74	0.23
Active	1 DCO, closed-loop	All banks on	2.41	1.61	0.23
Active	1 DCO, closed-loop	Single bank on	2.32	1.55	0.23
Inactive	1 DCO, closed-loop	Single bank on	1.57	1.36	0.23

8.6.3 VDDIO Power

The significant contributors to power drawn on VDDIO are the different I/Os of GAP9 used in output mode. Their contributions are highly dependent on the external load, and on their configuration, in particular the configured drive strength (see PADCFG* registers).

The following table provides the typical power consumption of an I/O configured in output mode, driving a PWM signal (50% duty cycle) to a 14pF external load.

I/O type	Always-on					
Drive	2mA		4mA		8mA	
Frequency (MHz)	Power (mW)	Energy per toggling (pJ)	Power (mW)	Energy per toggling (pJ)	Power (mW)	Energy per toggling (pJ)
0.5	0.06	59.1	0.08	82.8	0.08	75.6
1	0.12	61.2	0.13	64.8	0.13	64.1
3	0.37	62.4	0.37	62.4	0.35	58.8
5	0.61	61.2	0.59	59.0	0.62	61.9
10	1.35	67.6	1.34	66.8	1.35	67.5
20	3.06	76.5	2.95	73.8	2.94	73.5
30	3.88	64.6	3.93	65.6	3.97	66.1
40	3.62	45.2	3.98	49.8	4.22	52.7
50	4.37	43.7	4.37	43.7	4.37	43.7

I/O type	Switchable (non memory interface)					
Drive	2mA		4mA		8mA	
Frequency (MHz)	Power (mW)	Energy per toggling (pJ)	Power (mW)	Energy per toggling (pJ)	Power (mW)	Energy per toggling (pJ)
0.5	0.04	39.6	0.07	72.0	0.07	72.0
1	0.10	48.6	0.12	59.4	0.10	50.4
3	0.27	45.6	0.34	56.2	0.29	48.6
5	0.58	57.6	0.50	50.4	0.51	51.5
10	1.22	61.2	1.23	61.4	1.28	64.1
20	2.06	51.5	1.99	49.7	1.91	47.7
30	3.21	53.5	2.95	49.1	2.84	47.4
40	5.53	69.1	6.29	78.7	6.18	77.3
50	5.80	58.0	8.44	84.4	9.65	96.5

8.7 Boot and Wake-Up Times

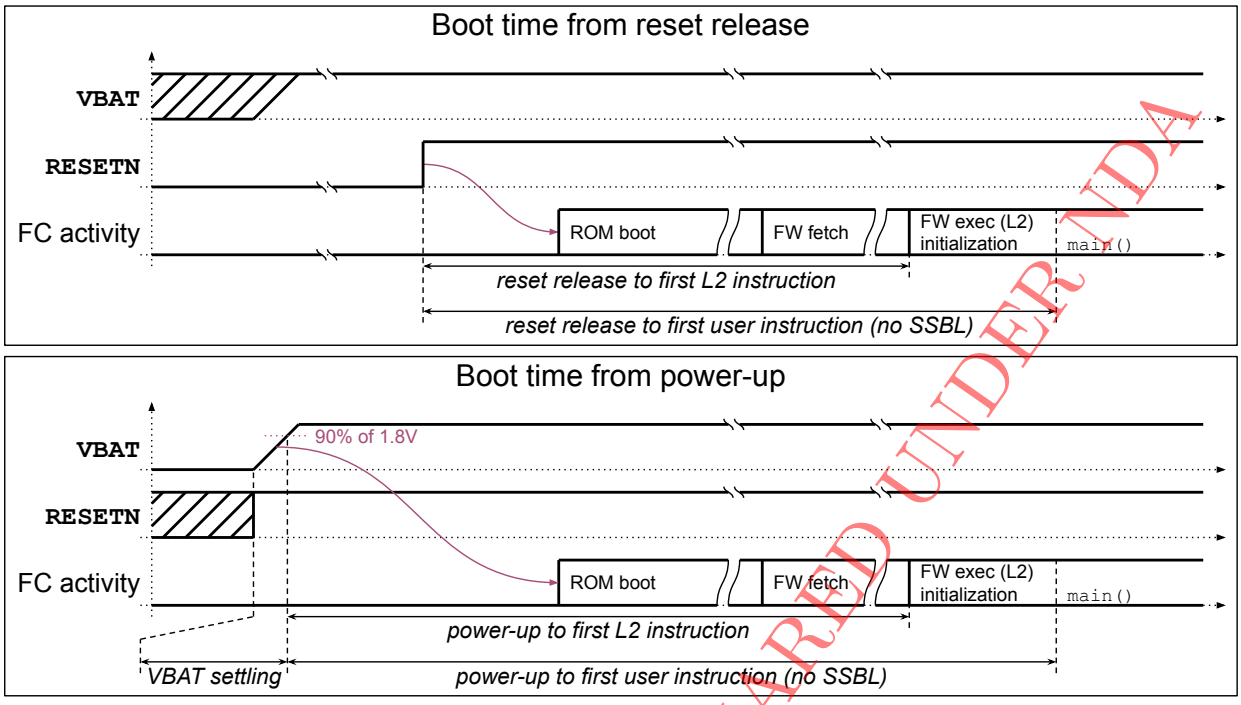
As presented in the Boot Process chapter, GAP9 boot and wake-up times involve the starting of a number hardware blocks, then execution of software, first from the boot ROM and then the firmware (typically fetched from MRAM or an external flash memory).

The startup time of hardware blocks depends on parameters such as supply ramp-up time or the use of the external reset signal.

The time spent in executing the ROM boot code may vary a lot depending on the boot configuration stored in the eFuses, for instance if the FLL needs to be configured in closed-loop mode or if the FC I-cache is activated.

Then the time required to fetch the firmware and start its execution is highly dependent on the selected boot device, on the size of the firmware initialization code and static data, on whether the XIP is used or not (and its page configuration), etc. Note that depending on the target system and application, the firmware can directly be the end application's binary code, or a second-stage boot loader (SSBL), which in turn fetches the end application's binary code.

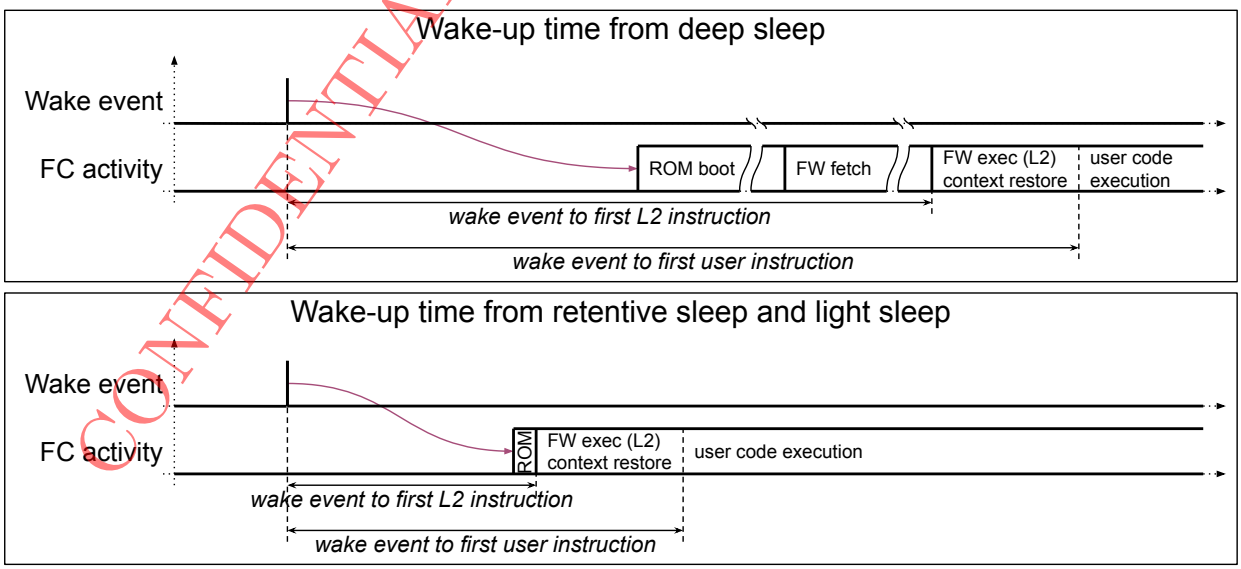
The next picture shows the different boot phases, and the following table gives typical boot duration in different configurations. For measures relying on power-on reset, VBAT is 1.8V, and its settling time is ~500µs.



Configured eFuses	Code size	Boot device	XIP pages	FLL configuration	Reset to FW	Reset to main()	Power-up to FW	Power-up to main()
None	32kB	MRAM	Unused	Open-loop	1.61ms	1.80ms	2.31ms	2.50ms
None	32kB	MRAM	16×512B	Open-loop	1.56ms	1.80ms	2.26ms	2.50ms
None	32kB	MRAM	Unused	Closed-loop	1.61ms	2.48ms	2.31ms	3.18ms
None	32kB	MRAM	16×512B	Closed-loop	1.56ms	2.54ms	2.26ms	3.24ms

The next picture illustrates the wake-up times from deep sleep, and retentive and light sleeps. For the former, the ROM is in charge of initializing again the chip to retrieve firmware and resume execution, whereas for both retentive sleep and light sleep, the ROM boot mainly aims at jumping into the firmware routines saved in retentive L2 memory. Note that since the context restore duration is highly dependent on the application.

The following table gives typical wake-up duration until FW execution in L2, depending on the sleep mode.



Sleep mode	Wake event to FW execution
Deep sleep	1.45ms
Retentive sleep	255us (TBC)
Light sleep	344us (TBC)

Chapter 9

Revisions

Version	Date	Changes
v0.1	2021/10/11	Initial release
v0.2	2021/10/24	Restructured document organization and fixed TOC generation Added introduction Added Quiddikey and timestamping descriptions Added reset values in register descriptions Minor fixes and improvements throughout the document
v0.3	2021/12/09	Major update: Proofread all devices, added missing cluster event unit and added details for devices like SFU, SAI, NE16, etc. Added address map pictures, fixed mistakes and added cross references to register descriptions Reorganized device section (grouped cluster devices, reordered according to address map Added confidentiality watermark Various improvements throughout the document
v0.4	2022/02/15	Improved formating (titles, headers and footers) Register offset is now in hexadecimal format Minor fixes
v0.5	2022/03/15	Added version table with change log Fixed mistakes in system clocks drawing and added details on FLL in closed loop mode Added description of the cluster I-cache Added constraints on SPI slave clocking Removed polarity inversion for memory controllers Minor additions: version of MIPI CSI-2, missing UART register, I/0 index in the WLCSP ballout description, precisions on the use of an external FAST REF clock Minor fixes
v0.6	2022/04/15	Added “Operating conditions” section Fixed I2C clocking diagram Removed BOR Fixed error in MRAM clock divider register Minor fixes throughout the document
v0.7	2022/07/01	New diagrams and clarifications in SAI section Corrected events and interrupt tables Improved description of pad configuration and fixed error in description of wake-up SPI selection Minor fixes throughout the document

Version	Date	Changes
v0.8	2022/08/30	Added setup/hold/output delay table for SAI; corrected maximum frequency values Corrected section on Memory Protection Unit, which was not aligned with the hardware version Additions and clarifications in Operating Conditions section Minor fixes throughout the document
v0.9	2022/09/30	Fixed size of uDMA ID in SAI and CPI registers Added Local Supply Monitoring section Corrected description of watchdog registers Added new power and boot time data in Operating Conditions Minor fixes throughout the document
v0.10	2022/11/28	Added MRAM reliability, wake-up times data and light sleep power in Operating Conditions Improved the description of the CSI-2 clocking scheme and contraints Added the description of the hardware limitation of reprogrammable I/Os Fixed L1 size in memory map table and diagram Removed description of uDMA filter unit Added details and cross references in the description of power domains Fixed range of CSI_VCCA065 in Power Supplies figure Updated Introduction, added bandwidths diagram Clearly identified always-on registers of SoC Controller Minor fixes throughout the document