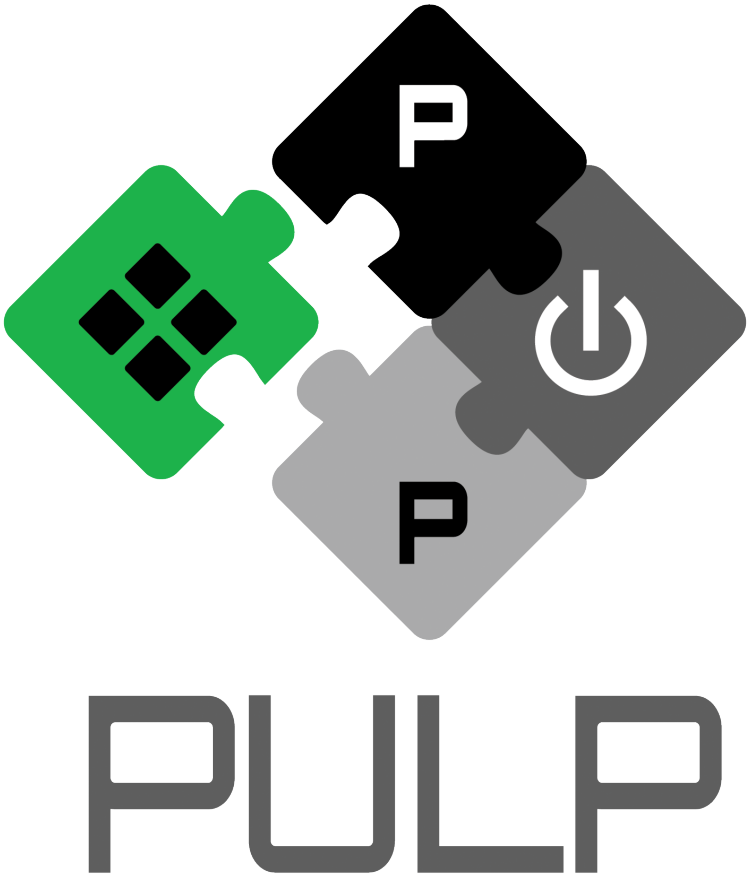
PULPissimo: Datasheet



The PULP team
pulp-info@list.ee.ethz.ch

May 11, 2020

Contents

1 Overview

PULPissimo is a 32 bit RI5CY single-core System-on-a-Chip. PULPissimo is the second version of the PULPino system and it can be extended with the multi-core cluster of the PULP project.

Differently from the simpler PULPino system, PULPissimo uses a more complex memory subsystem, an autonoumous I/O subsystem which uses the uDMA, new peripherals (eg the camera interface) and a new SDK.

Figure **??** shows a simplified block diagram of the SoC. As for PULPino, PULPissimo can be configured at design time to use either the RISC-V or zero-riscy. The peripherals are connected to the uDMA which transfers the date to the memory subsystem efficiently. The JTAG and the AXI plug have also access to the SoC. The AXI plug can be used to extend the microcontroller with a multi-core cluster or an accelerator. As for PULPino, the advanced debug unit is used to access to system and core registers, memories and memory-mapped IO via JTAG. A logarithmic interconnect allows to link the core and the uDMA to the memory banks simultaneously.

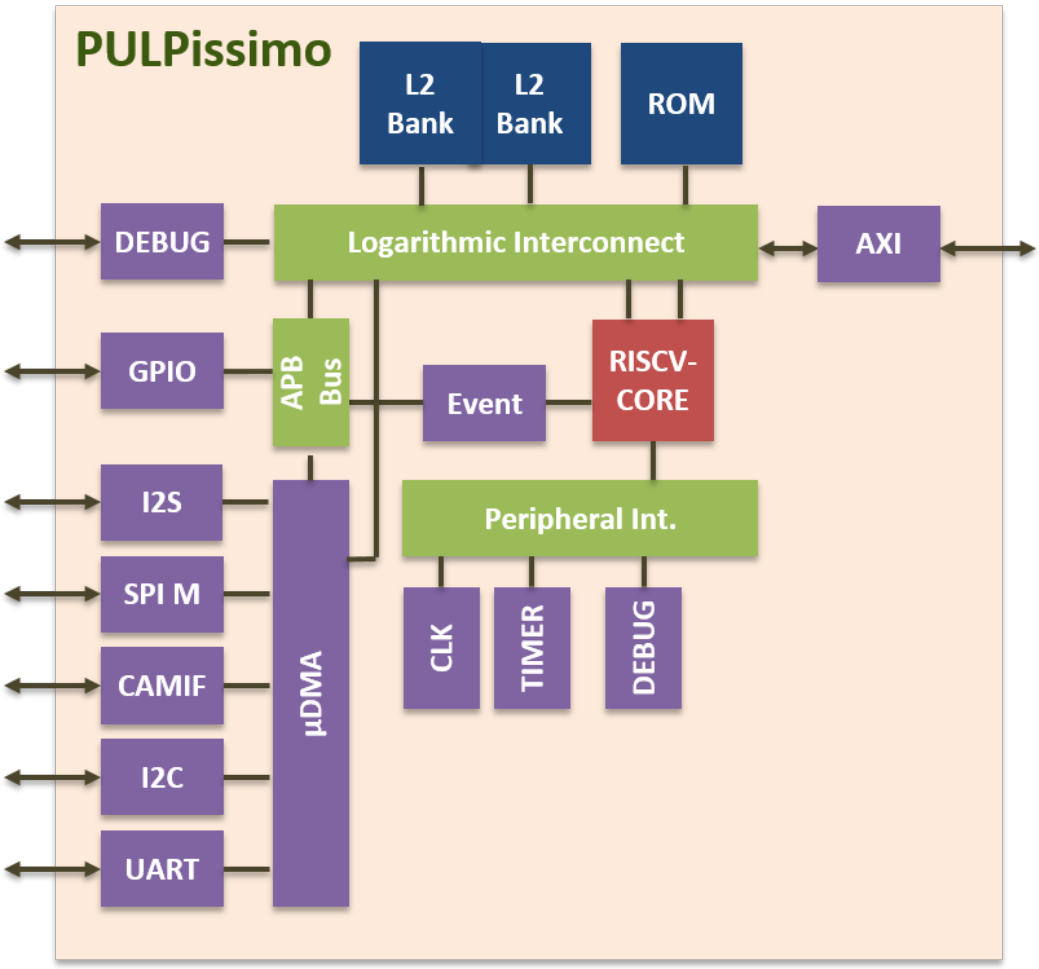


Figure 1.1: PULPissimo Overview.

PULPissimo is mainly targeted at RTL simulation and ASICs. The FPGA versions has not yet been implemented.

2 Memory Map

Figure **??** shows the default memory-map of PULPissimo, whereas Please, consult the uDMA documentation for the peripherals attached to the uDMA memory-map of configuration.

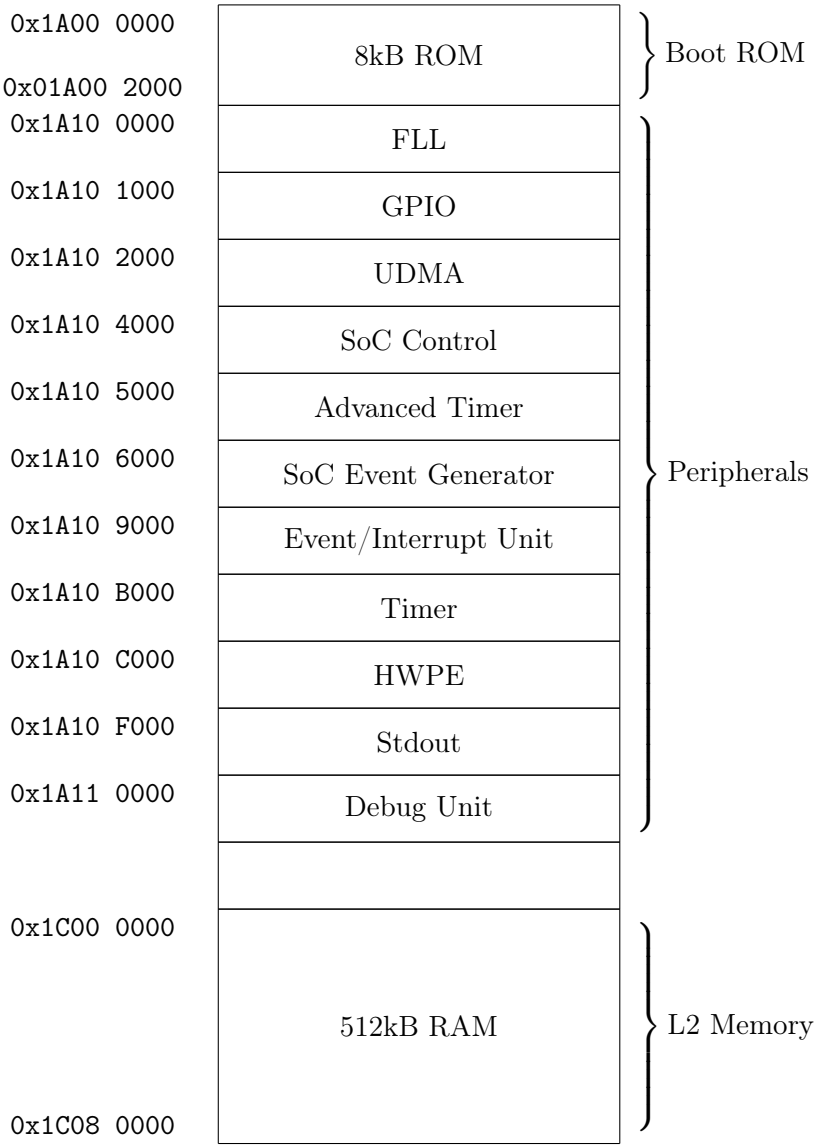


Figure 2.1: PULPissimo memory-map.

3 CPU Core

PULPissimo supports both the RISC-V and the zero-riscy RI5CY core. The two cores have the same external interfaces and are thus plug-compatible. Figure **??** and **??** show the two cores architectures.

For debugging purposes, all core registers have been memory mapped which allows to them to be accessed over the logaritmic-interconnect subsystem. The debug unit inside the core handles the request over this bus and reads/sets the core registers and/or halts the core.

The core supports performance counters. Those are mainly used for counting core internal events like stalls, but it is possible to count core-external events as well. For this purpose there is the `ext_perf_counters_i` port where arbitrary events can be attached. The core then increases its internal performance counter for this event type every time a logic high is seen on this port.

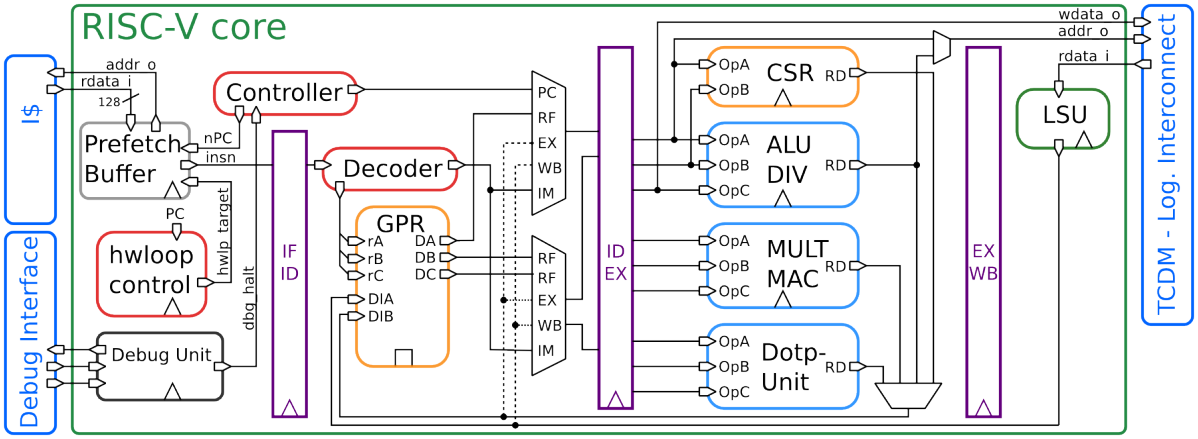


Figure 3.1: RISCY core overview

Take a look at the cores documentation for more details.

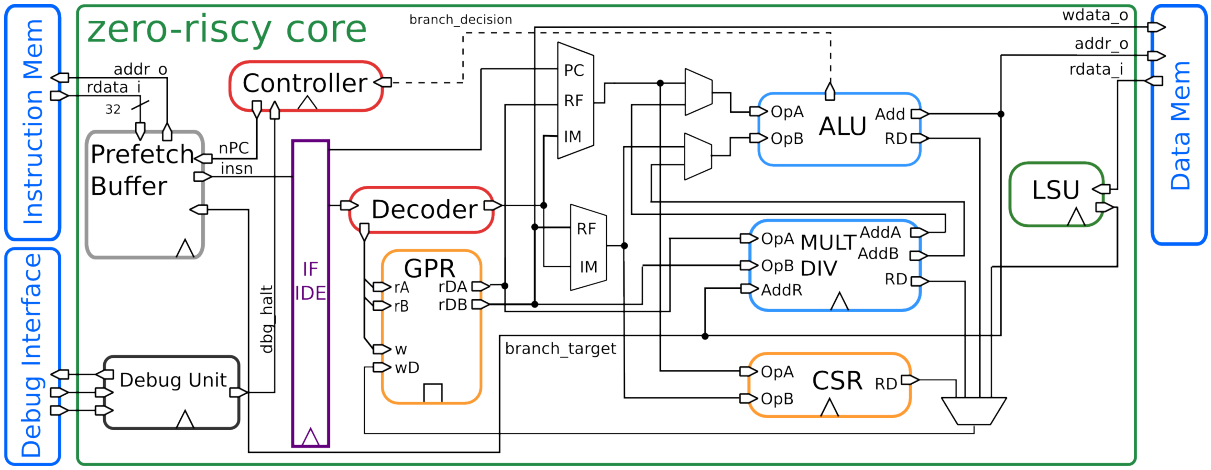


Figure 3.2: zero-riscy core overview

4 Peripherals

Most of the peripherals in PULPissimo are connected to the uDMA subsystem which efficiently handles all the data-transfers autonoumsly. The uDMA must be programmed by the core via memory-mapped read and write operations to receive commands.

See the uDMA documentation for more details under the uDMA repository.

The GPIO, timers, event unit and event generator, debug and the FLLs are not connected to the uDMA instead but to the APB bus. Following a brief overview about these units is given.

4.1 FLL

PULPissimo containts 3 FLLs. One FLL is meant for generating the clock for the peripheral domain, one for the core domain (core, memories, event unit etc) and one is meant for the cluster. The latter is not used.

All the FLLs can be bypassed by writing to the JTAG register before the reset signal is asserted. See Section **??** for more details about the bypass register.

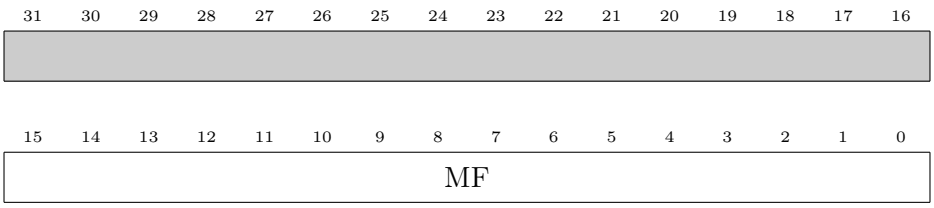
4.1.1 SoC FLL registers

Name	Address	Size	Type	Access	Default	Description
STATUS	0x1A100000	32	Status	R	0x00000000	FLL status register
CFG1	0x1A100004	32	Config	R/W	0x00000000	FLL configuration 1 register
CFG2	0x1A100008	32	Config	R/W	0x00000000	FLL configuration 2 register
INTEG	0x1A10000C	32	Config	R/W	0x00000000	FLL integrator configuration register.

Table 4.2: SoC FLL register table

4.1.2 STATUS

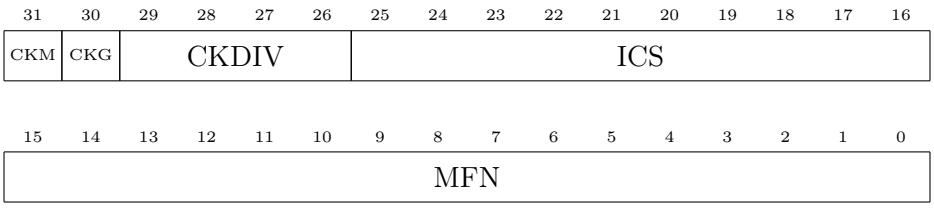
Address: 0x1A10_0000
Reset Value: 0x0000_0000



Bit 15-0 **MF** *(R)* Current DCO multiplication factor value bitfield

4.1.3 CFG1

Address: 0x1A10_0004
Reset Value: 0x0000_0000



`Bit 31` **CKM** *(R/W)* FLL operation mode configuration bitfield

- 0b0: standalone
- 0b1: normal

`Bit 30` **CKG** *(R/W)* FLL output clock divider configuration

- 0b0: not gated
- 0b1: gated

`Bit 29-26` **CKDIV** *(R/W)* FLL output clock divider configuration

`Bit 25-16` **ICS** *(R/W)* DCO input code in standalone

`Bit 15-0` **MFN** *(R/W)* Target clock multiplication factor in normal mode

4.1.4 CFG2

Address: `0x1A10_0008`
Reset Value: `0x0000_0000`

`Bit 31` **DITH** *(R/W)* Dithering activation

`Bit 30` **CKM** *(R/W)* Open loop when locked

- 0b0: disabled
- 0b1: enabled

`Bit 29` **CKSEL** *(R/W)* Configuration clock selection in standalone mode

- 0b0: DCO clock
- 0b1: Reference clock

`Bit 27-16` **LT** *(R/W)* Lock tolerance configuration. It is the margin around the multiplication factor within which the output clock is considered stable.

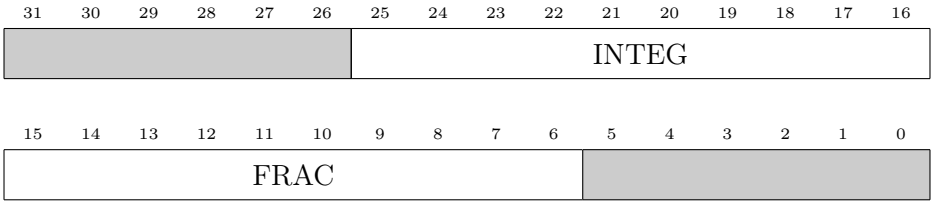
`Bit 15-10` **SCKL** *(R/W)* Number of stable REFCLK cycles until LOCK assert in normal mode. Uppper 6 bits of LOCK assert counter target in standalone mode.

`Bit 9-4` **UCKL** *(R/W)* Number of unstable REFCLK cycles until LOCK de-assert in normal mode. Lower 6 bits of LOCK assert counter target in standalone mode.

`Bit 3-0` **LG** *(R/W)* FLL loop gain setting

4.1.5 INTEG

Address: 0x1A10_000C
Reset Value: 0x0000_0000



Bit 25-16 **INTEG** *(R/W)* Integer part of integrator state bitfield. It corresponds to DCO unit bits.

Bit 15-6 **FRAC** *(R/W)* Fractional part of integrator state bitfield. It corresponds to dither unit input.

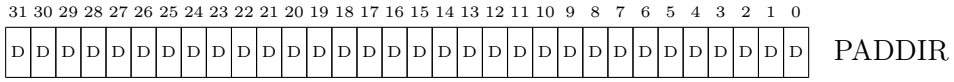
4.2 GPIO

Table 4.3: GPIO Signals

Signal	Direction	Description
`gpio_in[31:0]`	**input**	Transmit Data
`gpio_out[31:0]`	**output**	Receive Data
`gpio_dir[31:0]`	**output**	Request to Send
`gpio_padcfg[5:0][31:0]`	**output**	Pad Configuration
`interrupt`	**output**	Interrupt (Rise or Fall or Level)

4.2.1 PADDIR (Pad Direction)

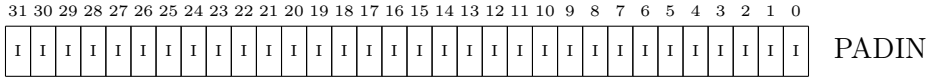
Address: `0x1A10_1000`
Reset Value: `0x0000_0000`



`Bit 31:0` **PADDIR**: Pad Direction.
Control the direction of each of the GPIO pads. A value of `1` means it is configured as an output, while `0` configures it as an input.

4.2.2 PADIN (Input Values)

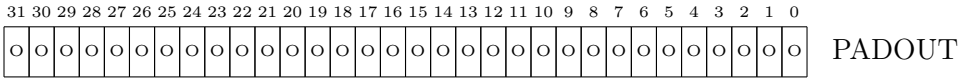
Address: `0x1A10_1004`
Reset Value: `0x0000_0000`



`Bit 31:0` **PADIN**: Input Values.

4.2.3 PADOUT (Output Values)

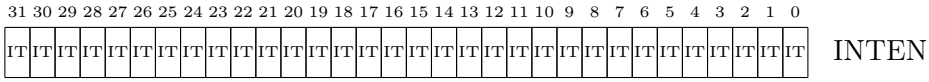
Address: `0x1A10_1008`
Reset Value: `0x0000_0000`



`Bit 31:0` **PADOUT**: Output Values.

4.2.4 INTEN (Interrupt Enable)

Address: `0x1A10_100C`
Reset Value: `0x0000_0000`



`Bit 31:0` **INTEN**: Interrupt Enable.
Interrupt enable per input bit. INTTYPE0 and INTTYPE1 control the interrupt triggering behavior.

There are four triggers available

- `INTTYPE0 = 0, INTTYPE1 = 0`: Level 1
- `INTTYPE0 = 1, INTTYPE1 = 0`: Level 0
- `INTTYPE0 = 0, INTTYPE1 = 1`: Rise
- `INTTYPE0 = 1, INTTYPE1 = 1`: Fall

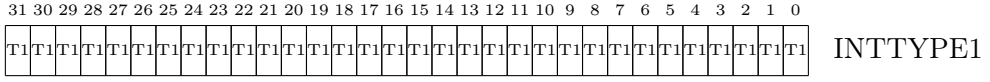
4.2.5 INTTYPE0 (Interrupt Type 0)

Address: `0x1A10_1010`
Reset Value: `0x0000_0000`

`Bit 31:0` **INTTYPE0**: Interrupt Type 0.
Controls the interrupt trigger behavior together with INTTYPE1. Use INTEN to enable interrupts first.

4.2.6 INTTYPE1 (Interrupt Type 1)

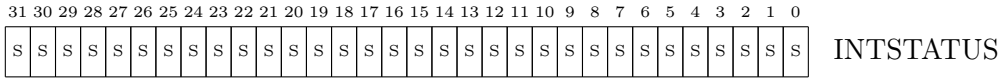
Address: `0x1A10_1014`
Reset Value: `0x0000_0000`



`Bit 31:0` **INTTYPE1**: Interrupt Type 1.
Controls the interrupt trigger behavior together with INTTYPE0. Use INTEN to enable interrupts first.

4.2.7 INTSTATUS (Interrupt Status)

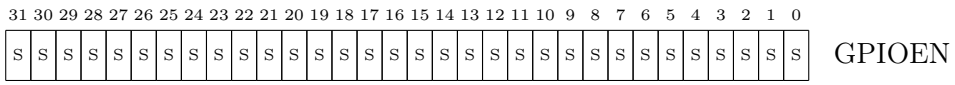
Address: `0x1A10_1018`
Reset Value: `0x0000_0000`



`Bit 31:0` **INTSTATUS**: Interrupt Status.
Contains interrupt status per GPIO line. The status register is cleared when read. Similarly the `interrupt` line is high while a bit is set in interrupt status and will be deasserted when the status register is read.

4.2.8 GPIOEN (GPIO Enable)

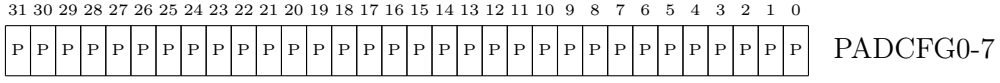
Address: `0x1A10_101C`
Reset Value: `0x0000_0000`



`Bit 31:0` **GPIOEN**: GPIO Enable.
Contains the enable bit per GPIO line.

4.2.9 PADCFG0-7 (Pad Configuration Registers 0-7)

Address: `0x1A10_1020 - 0x1A10_103C`
Reset Value: `0x0000_0000`



`Bit 31:0` **PADCFG0-7**: Pad Configuration Registers.
The pad configuration registers control various aspects of the pads that are typically used in ASICs, e.g. drive strength, Schmitt-Triggers, Slew Rate, etc. Since those configuration parameters depend on the exact pads used, each implementation is free to use the PADCFG0-7 registers in every way it wants and also leave them unconnected, if unneeded.

Writing to the PADOUTSET address (0x1A10_1040), the content of the PADOUT register is updated with its content "ored" with the write data.

Writing to the PADOUTCLR address (0x1A10_1044), the content of the PADOUT register is updated with its content "anded" with the inverted write data.

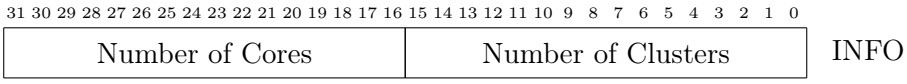
4.3 SoC Control

PULPissimo features a small and simple APB peripheral which provides information about the platform and provides the means for pad muxing on the ASIC.

The following registers can be accessed.

4.3.1 Info

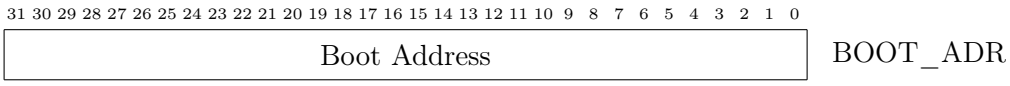
Address: `0x1A10_4000`
Reset Value: `0x0000_0000`



`Bit 31:0` **Info**: This register holds the number of clusters and the number of cores in the each cluster. It is a read-only register.

4.3.2 Boot Address

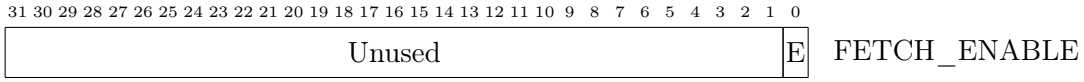
Address: `0x1A10_4004`
Reset Value: `0x1A10_0000`



`Bit 31:0` **Boot Address**: This register holds the boot address.

4.3.3 Fetch Enable

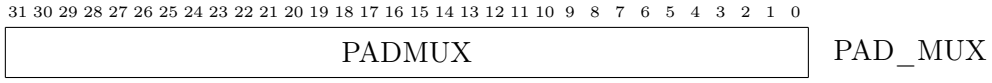
Address: `0x1A10_4008`
Reset Value: `0x0000_0001`



`Bit 31:0` **Fetch Enable**: This register contains the value of the fetch enable signal of the core.

4.3.4 PAD Mux

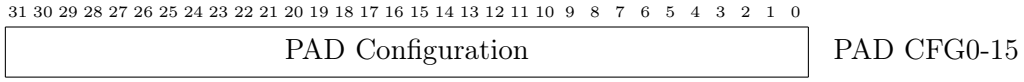
Address: `0x1A10_4010 - 0x1A10_401C`
Reset Value: `0x0000_0000`



`Bit 31:0` **PADMUX**: The content of these registers can be used to multiplex pads when targeting an ASIC. The first register (0x1A10_4010) can be used to sets the mux (2 bit select) from pin 0 (bits [1:0]) to 15 (bits [31:30]). The second register (0x1A10_4014) can be used to sets the mux (2 bit select) from pin 16 (bits [1:0]) to 31 (bits [31:30]). The third register (0x1A10_4018) can be used to sets the mux (2 bit select) from pin 32 (bits [1:0]) to 47 (bits [31:30]). The forth register (0x1A10_401C) can be used to sets the mux (2 bit select) from pin 48 (bits [1:0]) to 63 (bits [31:30]).

4.3.5 PAD Configuration

Address: `0x1A10_4020 - 0x1A10_405C`
Reset Value: `0x0000_0000`



`Bit 31:0` **PAD CFG0-15**: These 16 registers can be used for ASIC targets to configure pads, e.g. pull up, pull down values.

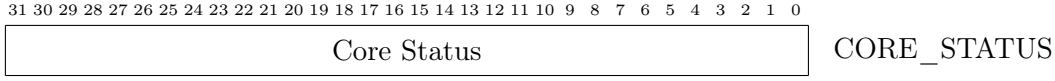
4.3.6 JTAG Register

Address: `0x1A10_4074`
Reset Value: `0x0000_0000`

`Bit 31:0` **JTAG Register**: This register contains the value of the input from the JTAG and can be used to write 8bit in the JTAG output register for system-to-JTAG communications.

4.3.7 Core Status

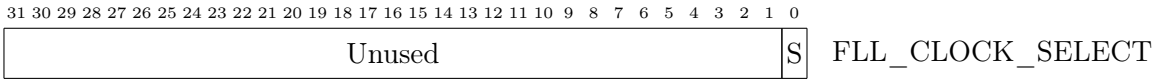
Address: `0x1A10_40A0 and 0x1A10_40C0`
Reset Value: `0x0000_0001`



`Bit 31:0` **Core Status**: These 2 registers contain the status of the system for testing/verification purposes like End Of Computation. The 0x1A10_40C0 register is read-only.

4.3.8 FLL Clock Select

Address: `0x1A10_40C8`
Reset Value: `0x0000_0000`



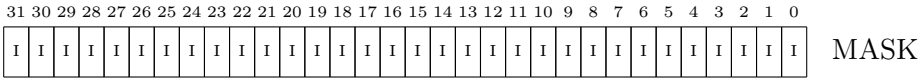
`Bit 31:0` **FLL Clock Select**: This register contains whether the system clock is coming from the FLL or the FLL is bypassed. It is a read-only register by the core but it can be written via JTAG.

4.4 Event/Interrupt Controller

PULPissimo features a lightweight event and interrupt controller which supports vectorized interrupts and events of up to 32 lines. It contains a FIFO of events from the peripherals or SW events. When an interrupt is ready and it is enabled (not masked), the unit sends the 5-bit ID to the core and the interrupt request line is raised up. If the core takes the interrupt, it replies with the ID of the interrupt taken and the acknowledge signal. The communication between the interrupt controller and the core is completly asynchronous. Note that the interrupt controller can change the interrupt ID anytime but it must rely on the ID sent by the core to know which interrupt has been taken. This is an important feature that covers the situation where a higher priority interrupt request prevent another one that has been already sent to the core. Depending on the core state and core interrupt enable, the interrupt can be accepted within a couple of clock cycles.

4.4.1 Mask

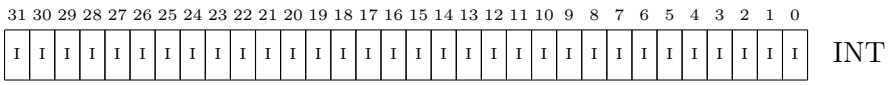
Address: `0x1A10_9000`
Reset Value: `0x0000_0000`



`Bit 31:0` **MASK**: This register contains the MASK (interrupt enable) for each of the 32 interrupts or events. Writing to 0x1A10_9004 sets the bits of the MASK register selected. Writing to 0x1A10_9008 clears the bits of the MASK register selected.

4.4.2 Interrupt

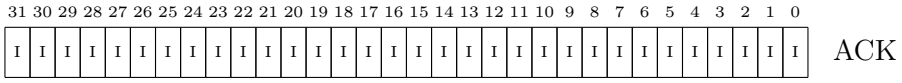
Address: `0x1A10_900C`
Reset Value: `0x0000_0000`



`Bit 31:0` **INT**: This register contains the pending interrupts or events. Writing to 0x1A10_9010 sets the bits of the INT register selected. Writing to 0x1A10_9014 clears the bits of the INT register selected.

4.4.3 Int Ack

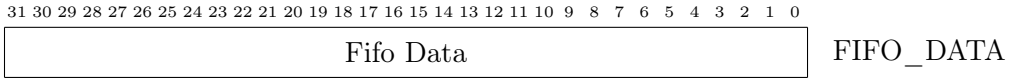
Address: `0x1A10_9018`
Reset Value: `0x0000_0000`



`Bit 31:0` **ACK**: This register contains the ACK (interrupt enable) for each of the 32 interrupts or events. Writing to 0x1A10_901C sets the bits of the ACK register selected. Writing to 0x1A10_9020 clears the bits of the ACK register selected.

4.4.4 FIFO Content

Address: `0x1A10_9024`
Reset Value: `0x0000_0000`



`Bit 31-0` **FIFO_DATA**: Fifo Content.
This is a read-only register that contain the first valid value of the FIFO.

4.5 SoC Event Generator

Events from peripherals and other sources can be forwarded to the fabric controller, cluster or (back) to certain peripherals, though for PULPissimo we don't have a cluster.

It is the SoC Event Generator's (`soc_event_generator.sv`) job to control which events are to be forwarded and where to. There are three set of masks available to do this:

FC Masks	Control which events are to be forwarded to the fabric controller
Cluster Masks	Control which events are to be forwarded to the cluster (disabled)
Peripheral Masks	Control which events are to be forwarded to peripherals

4.5.1 SoC Event Generator Registers

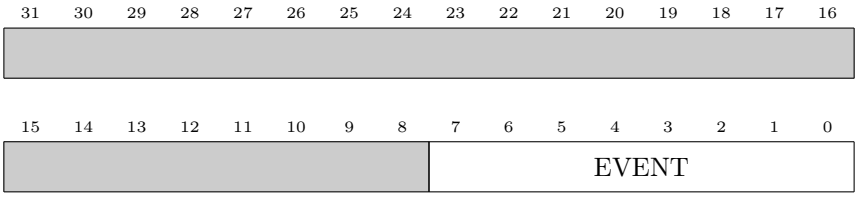
Name	Address	Size	Type	Access	Default	Description
SW_EVENT	0x1A106000	32	Config	W	0x00000000	SoC software events trigger register
FC_MASK0	0x1A106004	32	Config	R/W	0xFFFFFFFF	Events 0-31 dispatch mask to FC
FC_MASK1	0x1A106008	32	Config	R/W	0xFFFFFFFF	Events 32-63 dispatch mask to FC
FC_MASK2	0x1A10600C	32	Config	R/W	0xFFFFFFFF	Events 64-95 dispatch mask to FC
FC_MASK3	0x1A106010	32	Config	R/W	0xFFFFFFFF	Events 96-127 dispatch mask to FC
FC_MASK4	0x1A106014	32	Config	R/W	0xFFFFFFFF	Events 128-159 dispatch mask to FC
FC_MASK5	0x1A106018	32	Config	R/W	0xFFFFFFFF	Events 160-191 dispatch mask to FC
FC_MASK6	0x1A10601C	32	Config	R/W	0xFFFFFFFF	Events 191-223 dispatch mask to FC
FC_MASK7	0x1A106020	32	Config	R/W	0xFFFFFFFF	Events 224-255 dispatch mask to FC
PR_MASK0	0x1A106044	32	Config	R/W	0xFFFFFFFF	Events 0-31 dispatch mask to peripherals
PR_MASK1	0x1A106048	32	Config	R/W	0xFFFFFFFF	Events 32-63 dispatch mask to peripherals
PR_MASK2	0x1A10604C	32	Config	R/W	0xFFFFFFFF	Events 64-95 dispatch mask to peripherals
PR_MASK3	0x1A106050	32	Config	R/W	0xFFFFFFFF	Events 96-127 dispatch mask to peripherals
PR_MASK4	0x1A106054	32	Config	R/W	0xFFFFFFFF	Events 128-159 dispatch mask to peripherals
PR_MASK5	0x1A106058	32	Config	R/W	0xFFFFFFFF	Events 160-191 dispatch mask to peripherals
PR_MASK6	0x1A10605C	32	Config	R/W	0xFFFFFFFF	Events 191-223 dispatch mask to peripherals
PR_MASK7	0x1A106060	32	Config	R/W	0xFFFFFFFF	Events 224-255 dispatch mask to peripherals

ERR0	0x1A106064	32	Status	R	0x00000000	Events 0-31 event queue overflow
ERR1	0x1A106068	32	Status	R	0x00000000	Events 32-63 event queue overflow
ERR2	0x1A10606C	32	Status	R	0x00000000	Events 64-95 event queue overflow
ERR3	0x1A106070	32	Status	R	0x00000000	Events 96-127 event queue overflow
ERR4	0x1A106074	32	Status	R	0x00000000	Events 128-159 event queue overflow
ERR5	0x1A106078	32	Status	R	0x00000000	Events 160-191 event queue overflow
ERR6	0x1A10607C	32	Status	R	0x00000000	Events 191-223 event queue overflow
ERR7	0x1A106080	32	Status	R	0xFFFFFFFF	Events 224-255 event queue overflow
TIMER_LO	0x1A106084	32	Status	R/W	0xFFFFFFFF	Trigger Timer LO of APB Timer with event
TIMER_HI	0x1A106088	32	Status	R/W	0xFFFFFFFF	Trigger Timer HI of APB Timer with event

Table 4.5: SoC Event Generator register table

4.5.2 SW_EVENT

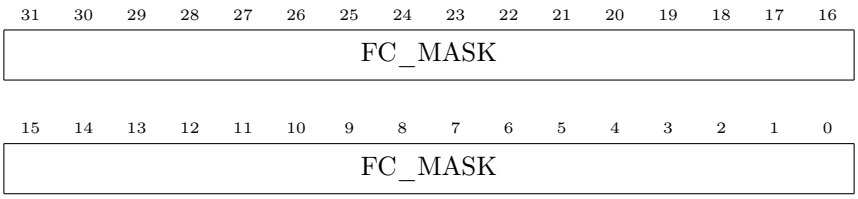
Address: 0x1A10_6000
Reset Value: 0x0000_0000



Bit 7-0 **EVENT** *(W)* Writing a one-hot value into EVENT triggers a SoC software event. 8 software events are available.

4.5.3 FC_MASKX, $X = 0...7$

Address: 0x1A10_6004 + 0x4 * X
Reset Value: 0xFFFF_FFFF

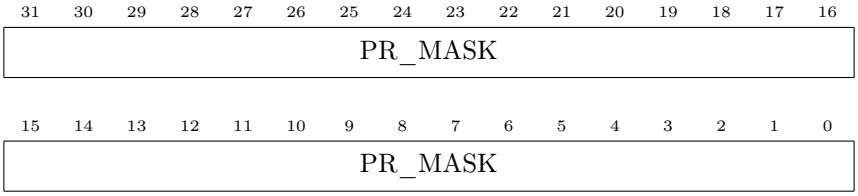


`Bit 31-0` **FC_MASK** *(R/W)* Event Mask to enable/disable event dispatch to FC interrupt controller.

- Setting $bit[i]$ to $0b1$ disables dispatching $event[32 * X + i]$ to FC interrupt controller.
- Setting $bit[i]$ to $0b0$ enables dispatching $event[32 * X + i]$ to FC interrupt controller.

4.5.4 PR_MASKX, $X = 0...7$

Address: `0x1A10_6044 + 0x4 *` X
Reset Value: `0xFFFF_FFFF`

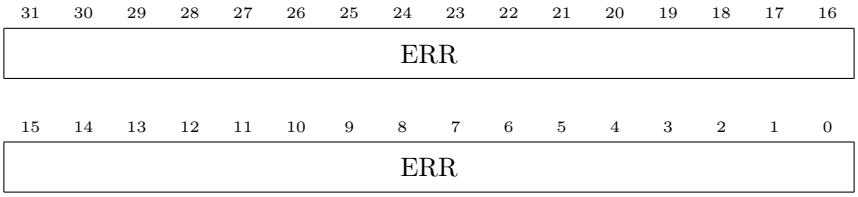


`Bit 31-0` **PR_MASK** *(R/W)* Event Mask to enable/disable event dispatch to peripherals.

- Setting $bit[i]$ to $0b1$ disables dispatching $event[32 * X + i]$ to peripherals.
- Setting $bit[i]$ to $0b0$ enables dispatching $event[32 * X + i]$ to peripherals.

4.5.5 ERRX, $X = 0...7$

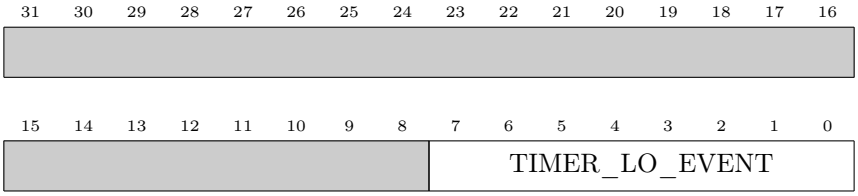
Address: `0x1A10_6064 + 0x4 *` X
Reset Value: `0x0000_0000`



`Bit 31-0` **ERR** *(R/W)* Event queue overflow. Clear after read. Reading $0b1$ at $ERR[i]$ means the event queue of event with id $32 * X + i$ overflowed.

4.5.6 TIMER_LO

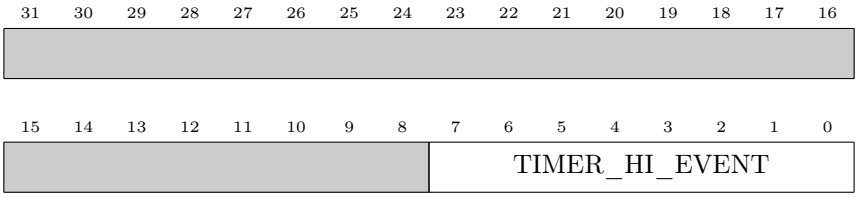
Address: `0x1A10_6084`
Reset Value: `0x0000_0000`



`Bit 7-0` **TIMER_LO_EVENT** *(R/W)* Trigger and start APB Timer LO by the event with id that equals TIMER_LO_EVENT

4.5.7 TIMER_HI

Address: `0x1A10_6088`
Reset Value: `0x0000_0000`



`Bit 7-0` **TIMER_HI_EVENT** *(R/W)* Trigger and start APB Timer HI by the event with id that equals TIMER_HI_EVENT

4.6 APB Timer

The APB Timer (`apb_timer_unit.sv`) has the following features:

- 2 general purpose 32-bit upwards counters
- Can be triggered by multiple sources:
  - FLL clock
  - FLL clock + Prescale
  - Reference clock at 32 kHz
  - Any event
- 8-bit programmable prescaler (divides the FLL clock frequency)
- Different counting modes:
  - One shot mode: timer is stopped after the first comparison match
  - Continuous mode: timer continues couting after a match
  - 64-bit cascaded mode: use both 32-bit timers as a 64-bit timer
- Interrupt request generation on comparison match

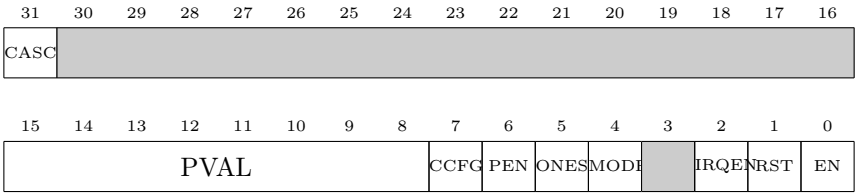
4.6.1 APB Timer Registers

Name	Address	Size	Type	Access	Default	Description
CFG_LO	0x1A10B000	32	Config	R/W	0x00000000	Timer Low Configuration register
CFG_HI	0x1A10B004	32	Config	R/W	0x00000000	Timer High Configuration register
CNT_LO	0x1A10B008	32	Data	R/W	0x00000000	Timer Low counter value register
CNT_HI	0x1A10B00C	32	Data	R/W	0x00000000	Timer High counter value register
CMP_LO	0x1A10B010	32	Config	R/W	0x00000000	Timer Low comparator value register
CMP_HI	0x1A10B014	32	Config	R/W	0x00000000	Timer High comparator value register
START_LO	0x1A10B018	32	Config	R/W	0x00000000	Start Timer Low counting register
START_HI	0x1A10B01C	32	Config	R/W	0x00000000	Start Timer High counting register
RESET_LO	0x1A10B020	32	Config	R/W	0x00000000	Reset Timer Low counter register
RESET_HI	0x1A10B024	32	Config	R/W	0x00000000	Reset Timer High counter register

Table 4.7: APB Timer register table

4.6.2 CFG_LO

Address: `0x1A10_B000`
Reset Value: `0x0000_0000`



`Bit 31` **CASC** *(R/W)* Timer low and Timer high 64-bit cascaded mode enable bit

`Bit 15-8` **PVAL** *(R/W)* Timer low prescaler value. $f_{timer} = f_{clk}/(1 + PVAL)$

`Bit 7` **CCFG** *(R/W)* Timer low clock source configuration

- $0b0$: FLL or FLL plus Prescaler
- $0b1$: 32 kHz reference clock

`Bit 6` **PEN** *(R/W)* Timer low prescaler enable bit

`Bit 5` **ONES** *(R/W)* Timer low one shot configuration

- $0b0$: Timer stays enabled after a compare match with CMP_LO
- $0b1$: Timer is disabled after a compare match with CMP_LO

`Bit 4` **MODE** *(R/W)* Timer low continuous mode configuration

- $0b0$: Continue incrementing timer low counter after a compare match with CMP_LO
- $0b1$: Reset timer to after a compare match with CMP_LO

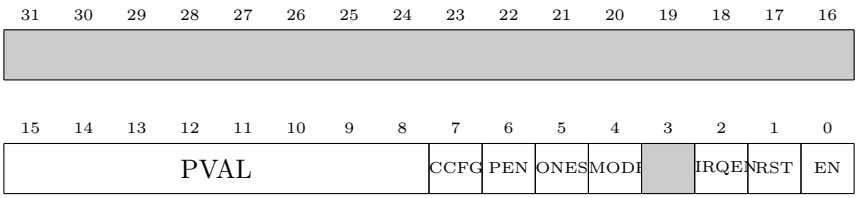
`Bit 2` **IRQEN** *(R/W)* Timer low interrupt generation on compare match enable

`Bit 1` **RST** *(R/W)* Timer low reset, cleared after the reset happened

`Bit 0` **EN** *(R/W)* Timer enable (starts couting) bit

4.6.3 CFG_HI

Address: `0x1A10_B004`
Reset Value: `0x0000_0000`



`Bit 16-8` **PVAL** *(R/W)* Timer hi prescaler value. $f_{timer} = f_{clk}/(1 + PVAL)$

`Bit 7` **CCFG** *(R/W)* Timer hi clock source configuration

- $0b0$: FLL or FLL plus Prescaler

- 0*b*1: 32 kHz reference clock

`Bit 6` **PEN** *(R/W)* Timer hi prescaler enable bit

`Bit 5` **ONES** *(R/W)* Timer hi one shot configuration

- 0*b*0: Timer stays enabled after a compare match with CMP_HI
- 0*b*1: Timer is disabled after a compare match with CMP_HI

`Bit 4` **MODE** *(R/W)* Timer hi continuous mode configuration

- 0*b*0: Continue incrementing timer hi counter after a compare match with CMP_HI
- 0*b*1: Reset timer to after a compare match with CMP_HI

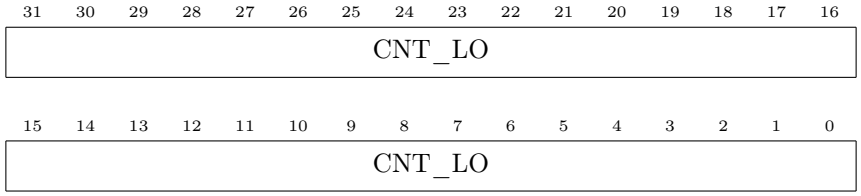
`Bit 2` **IRQEN** *(R/W)* Timer hi interrupt generation on compare match enable

`Bit 1` **RST** *(R/W)* Timer hi reset, cleared after the reset happened

`Bit 0` **EN** *(R/W)* Timer enable (starts couting) bit

4.6.4 CNT_LO

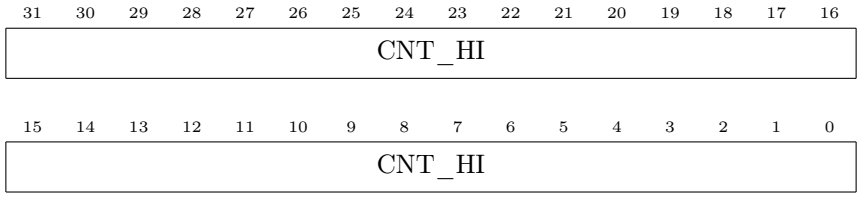
Address: `0x1A10_B008`
Reset Value: `0x0000_0000`



`Bit 31-0` **CNT_LO** *(R/W)* Timer low counter value

4.6.5 CNT_HI

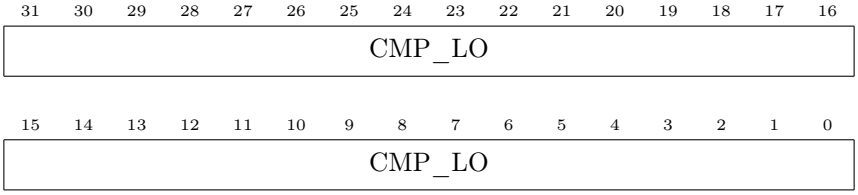
Address: `0x1A10_B00C`
Reset Value: `0x0000_0000`



`Bit 31-0` **CNT_HI** *(R/W)* Timer high counter value

4.6.6 CMP_LO

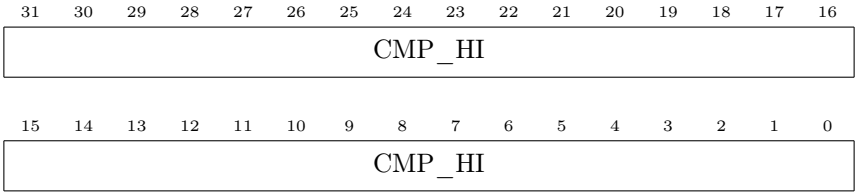
Address: `0x1A10_B010`
Reset Value: `0x0000_0000`



`Bit 31-0` **CMP_LO** *(R/W)* Timer low comparator value

4.6.7 CMP_HI

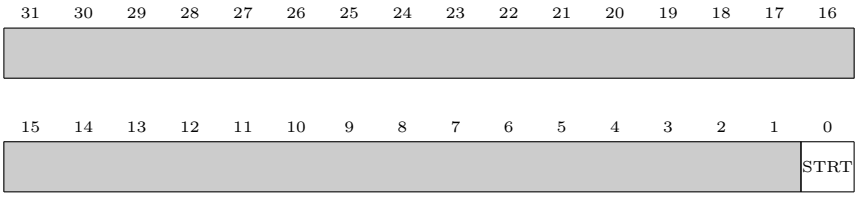
Address: `0x1A10_B014`
Reset Value: `0x0000_0000`



`Bit 31-0` **CMP_HI** *(R/W)* Timer high comparator value

4.6.8 START_LO

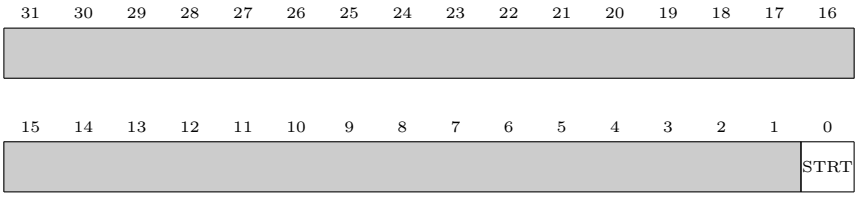
Address: `0x1A10_B018`
Reset Value: `0x0000_0000`



`Bit 0` **STRT** *(W)* Timer high start command (sets EN in CFG_LO)

4.6.9 START_HI

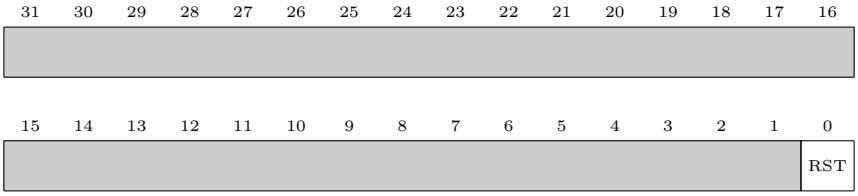
Address: `0x1A10_B01C`
Reset Value: `0x0000_0000`



`Bit 0` **STRT** *(W)* Timer high start command (sets EN in CFG_HI)

4.6.10 RESET_LO

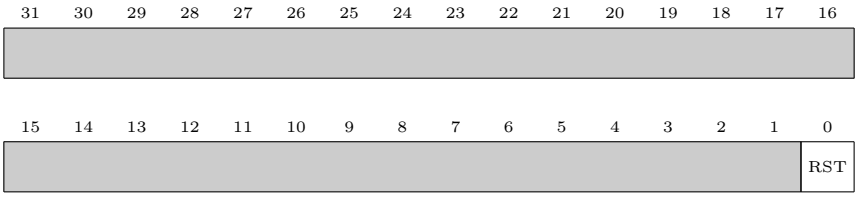
Address: `0x1A10_B020`
Reset Value: `0x0000_0000`



`Bit 0` **RST** *(W)* Timer high reset command (writes RST in CFG_LO)

4.6.11 RESET_HI

Address: `0x1A10_B024`
Reset Value: `0x0000_0000`



`Bit 0` **RST** *(W)* Timer high reset command (sets RST in CFG_HI)

4.7 APB Advanced Timer

4.7.1 APB Advanced Timer Registers

Name	Address	Size	Type	Access	Default	Description
T0_CMD	0x1A104000	32	Config	R/W	0x00000000	ADV_TIMER0 command register.
T0_CONFIG	0x1A104004	32	Config	R/W	0x00000000	ADV_TIMER0 configuration register.
T0_THRESHOLD	0x1A104008	32	Config	R/W	0x00000000	ADV_TIMER0 threshold configuration register.
T0_TH_CHANNEL0	0x1A10400C	32	Config	R/W	0x00000000	ADV_TIMER0 channel 0 threshold configuration register.
T0_TH_CHANNEL1	0x1A104010	32	Config	R/W	0x00000000	ADV_TIMER0 channel 1 threshold configuration register.
T0_TH_CHANNEL2	0x1A104014	32	Config	R/W	0x00000000	ADV_TIMER0 channel 2 threshold configuration register.
T0_TH_CHANNEL3	0x1A104018	32	Config	R/W	0x00000000	ADV_TIMER0 channel 3 threshold configuration register.
T0_COUNTER	0x1A10402C	32	Status	R	0x00000000	ADV_TIMER0 counter register.
T1_CMD	0x1A104040	32	Config	R/W	0x00000000	ADV_TIMER1 command register.
T1_CONFIG	0x1A104044	32	Config	R/W	0x00000000	ADV_TIMER1 configuration register.
T1_THRESHOLD	0x1A104048	32	Config	R/W	0x00000000	ADV_TIMER1 threshold configuration register.
T1_TH_CHANNEL0	0x1A10404C	32	Config	R/W	0x00000000	ADV_TIMER1 channel 0 threshold configuration register.
T1_TH_CHANNEL1	0x1A104050	32	Config	R/W	0x00000000	ADV_TIMER1 channel 1 threshold configuration register.
T1_TH_CHANNEL2	0x1A104054	32	Config	R/W	0x00000000	ADV_TIMER1 channel 2 threshold configuration register.

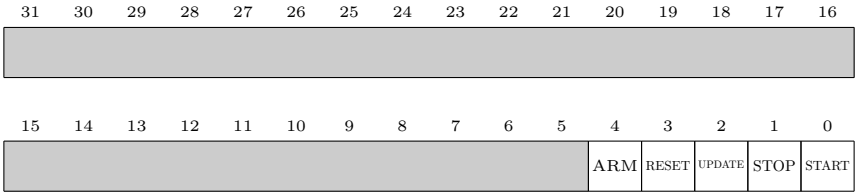
T1_TH_CHANNEL3	0x1A104058	32	Config	R/W	0x00000000	ADV_TIMER1 channel 3 threshold configuration register.
T1_COUNTER	0x1A10406C	32	Status	R	0x00000000	ADV_TIMER1 counter register.
T2_CMD	0x1A104080	32	Config	R/W	0x00000000	ADV_TIMER2 command register.
T2_CONFIG	0x1A104084	32	Config	R/W	0x00000000	ADV_TIMER2 configuration register.
T2_THRESHOLD	0x1A104088	32	Config	R/W	0x00000000	ADV_TIMER2 threshold configuration register.
T2_TH_CHANNEL0	0x1A10408C	32	Config	R/W	0x00000000	ADV_TIMER2 channel 0 threshold configuration register.
T2_TH_CHANNEL1	0x1A104090	32	Config	R/W	0x00000000	ADV_TIMER2 channel 1 threshold configuration register.
T2_TH_CHANNEL2	0x1A104094	32	Config	R/W	0x00000000	ADV_TIMER2 channel 2 threshold configuration register.
T2_TH_CHANNEL3	0x1A104098	32	Config	R/W	0x00000000	ADV_TIMER2 channel 3 threshold configuration register.
T2_COUNTER	0x1A1040AC	32	Status	R	0x00000000	ADV_TIMER2 counter register.
T3_CMD	0x1A1040C0	32	Config	R/W	0x00000000	ADV_TIMER3 command register.
T3_CONFIG	0x1A1040C4	32	Config	R/W	0x00000000	ADV_TIMER3 configuration register.
T3_THRESHOLD	0x1A1040C8	32	Config	R/W	0x00000000	ADV_TIMER3 threshold configuration register.
T3_TH_CHANNEL0	0x1A1040CC	32	Config	R/W	0x00000000	ADV_TIMER3 channel 0 threshold configuration register.
T3_TH_CHANNEL1	0x1A1040D0	32	Config	R/W	0x00000000	ADV_TIMER3 channel 1 threshold configuration register.
T3_TH_CHANNEL2	0x1A1040D4	32	Config	R/W	0x00000000	ADV_TIMER3 channel 2 threshold configuration register.

T3_TH_CHANNEL3	0x1A1040D8	32	Config	R/W	0x00000000	ADV_TIMER3 channel 3 threshold configuration register.
T3_COUNTER	0x1A1040EC	32	Status	R	0x00000000	ADV_TIMER3 counter register.
EVENT_CFG	0x1A104100	32	Config	R/W	0x00000000	ADV_TIMERS events configuration register.
CG	0x1A104104	32	Config	R/W	0x00000000	ADV_TIMERS channels clock gating configuration register.

Table 4.8: APB Advanced Timer

4.7.2 T0_CMD

Address: 0x1A104000
Reset Value: 0x00000000



Bit 4 **ARM** *(R/W)* ADV_TIMER0 arm command bitfield.

Bit 3 **RESET** *(R/W)* ADV_TIMER0 reset command bitfield.

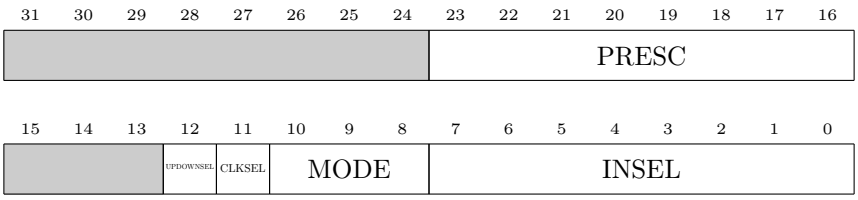
Bit 2 **UPDATE** *(R/W)* ADV_TIMER0 update command bitfield.

Bit 1 **STOP** *(R/W)* ADV_TIMER0 stop command bitfield.

Bit 0 **START** *(R/W)* ADV_TIMER0 start command bitfield.

4.7.3 T0_CONFIG

Address: 0x1A104004
Reset Value: 0x00000000



Bit 23 - 16 **PRESC** *(R/W)* ADV_TIMER0 prescaler value configuration bitfield.

`Bit 12` **UPDOWNSEL** *(R/W)* ADV_TIMER0 center-aligned mode configuration bitfield:
- 1'b0: The counter counts up and down alternatively.
- 1'b1: The counter counts up and resets to 0 when reach threshold.

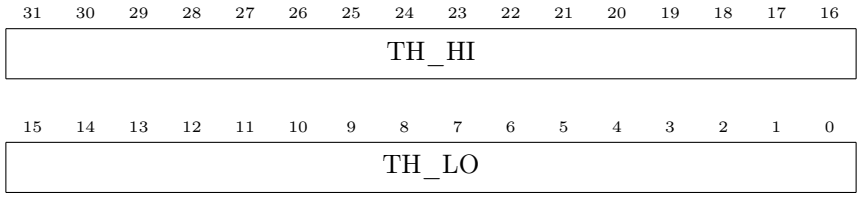
`Bit 11` **CLKSEL** *(R/W)* ADV_TIMER0 clock source configuration bitfield:
- 1'b0: FLL
- 1'b1: reference clock at 32kHz

`Bit 10 - 8` **MODE** *(R/W)* ADV_TIMER0 trigger mode configuration bitfield:
- 3'h0: trigger event at each clock cycle.
- 3'h1: trigger event if input source is 0
- 3'h2: trigger event if input source is 1
- 3'h3: trigger event on input source rising edge
- 3'h4: trigger event on input source falling edge
- 3'h5: trigger event on input source falling or rising edge
- 3'h6: trigger event on input source rising edge when armed
- 3'h7: trigger event on input source falling edge when armed

`Bit 7 - 0` **INSEL** *(R/W)* ADV_TIMER0 input source configuration bitfield:
- 0-31: GPIO[0] to GPIO[31]
- 32-35: Channel 0 to 3 of ADV_TIMER0
- 36-39: Channel 0 to 3 of ADV_TIMER1
- 40-43: Channel 0 to 3 of ADV_TIMER2
- 44-47: Channel 0 to 3 of ADV_TIMER3

4.7.4 T0_THRESHOLD

Address: `0x1A104008`
Reset Value: `0x00000000`

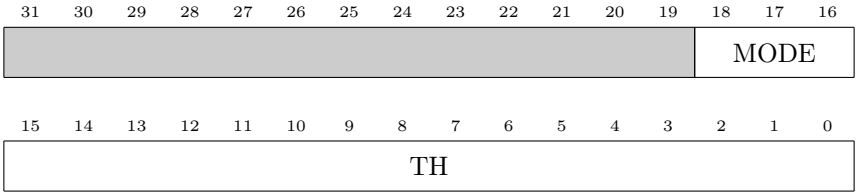


`Bit 31 - 16` **TH_HI** *(R/W)* ADV_TIMER0 threshold high part configuration bitfield. It defines end counter value.

`Bit 15 - 0` **TH_LO** *(R/W)* ADV_TIMER0 threshold low part configuration bitfield. It defines start counter value.

4.7.5 T0_TH_CHANNEL0

Address: `0x1A10400C`
Reset Value: `0x00000000`

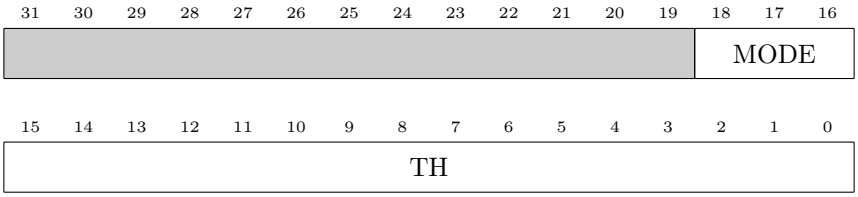


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER0 channel 0 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER0 channel 0 threshold configuration bitfield.

4.7.6 T0_TH_CHANNEL1

Address: `0x1A104010`
Reset Value: `0x00000000`

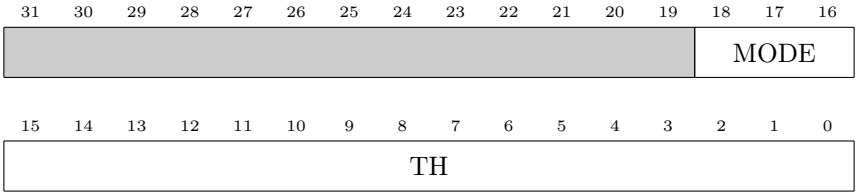


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER0 channel 1 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER0 channel 1 threshold configuration bitfield.

4.7.7 T0_TH_CHANNEL2

Address: `0x1A104014`
Reset Value: `0x00000000`

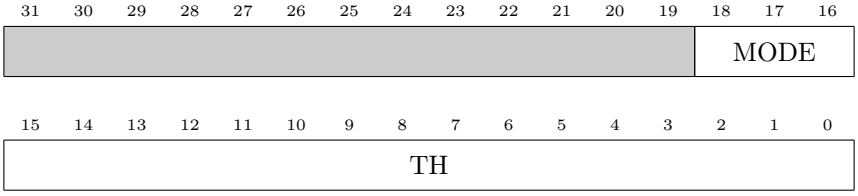


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER0 channel 2 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER0 channel 2 threshold configuration bitfield.

4.7.8 T0_TH_CHANNEL3

Address: `0x1A104018`
Reset Value: `0x00000000`

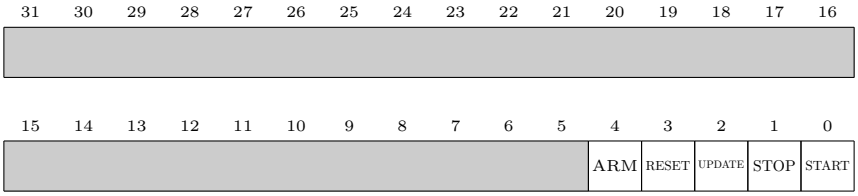


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER0 channel 3 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER0 channel 3 threshold configuration bitfield.

4.7.9 T1_CMD

Address: `0x1A104040`
Reset Value: `0x00000000`



`Bit 4` **ARM** *(R/W)* ADV_TIMER1 arm command bitfield.

`Bit 3` **RESET** *(R/W)* ADV_TIMER1 reset command bitfield.

`Bit 2` **UPDATE** *(R/W)* ADV_TIMER1 update command bitfield.

`Bit 1` **STOP** *(R/W)* ADV_TIMER1 stop command bitfield.

`Bit 0` **START** *(R/W)* ADV_TIMER1 start command bitfield.

4.7.10 T1_CONFIG

Address: `0x1A104044`
Reset Value: `0x00000000`

`Bit 23 - 16` **PRESC** *(R/W)* ADV_TIMER1 prescaler value configuration bitfield.

`Bit 12` **UPDOWNSEL** *(R/W)* ADV_TIMER1 center-aligned mode configuration bitfield:
- 1'b0: The counter counts up and down alternatively.
- 1'b1: The counter counts up and resets to 0 when reach threshold.

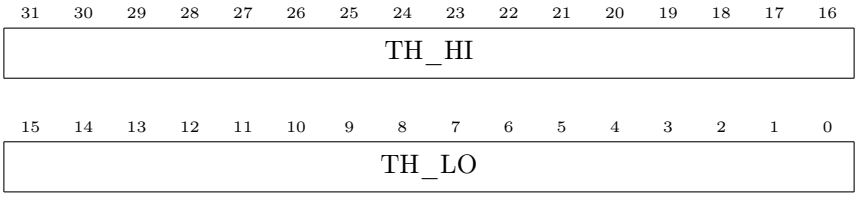
`Bit 11` **CLKSEL** *(R/W)* ADV_TIMER1 clock source configuration bitfield:
- 1'b0: FLL
- 1'b1: reference clock at 32kHz

`Bit 10 - 8` **MODE** *(R/W)* ADV_TIMER1 trigger mode configuration bitfield:
- 3'h0: trigger event at each clock cycle.
- 3'h1: trigger event if input source is 0
- 3'h2: trigger event if input source is 1
- 3'h3: trigger event on input source rising edge
- 3'h4: trigger event on input source falling edge
- 3'h5: trigger event on input source falling or rising edge
- 3'h6: trigger event on input source rising edge when armed
- 3'h7: trigger event on input source falling edge when armed

`Bit 7 - 0` **INSEL** *(R/W)* ADV_TIMER1 input source configuration bitfield:
- 0-31: GPIO[0] to GPIO[31]
- 32-35: Channel 0 to 3 of ADV_TIMER0
- 36-39: Channel 0 to 3 of ADV_TIMER1
- 40-43: Channel 0 to 3 of ADV_TIMER2
- 44-47: Channel 0 to 3 of ADV_TIMER3

4.7.11 T1_THRESHOLD

Address: 0x1A104048
Reset Value: 0x00000000

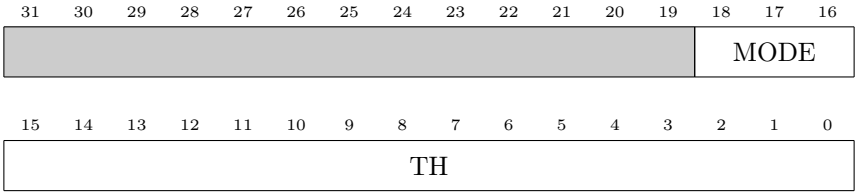


Bit 31 - 16 **TH_HI** *(R/W)* ADV_TIMER1 threshold high part configuration bitfield. It defines end counter value.

Bit 15 - 0 **TH_LO** *(R/W)* ADV_TIMER1 threshold low part configuration bitfield. It defines start counter value.

4.7.12 T1_TH_CHANNEL0

Address: 0x1A10404C
Reset Value: 0x00000000

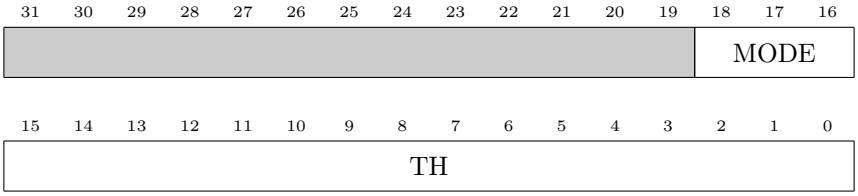


Bit 18 - 16 **MODE** *(R/W)* ADV_TIMER1 channel 0 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

Bit 15 - 0 **TH** *(R/W)* ADV_TIMER1 channel 0 threshold configuration bitfield.

4.7.13 T1_TH_CHANNEL1

Address: 0x1A104050
Reset Value: 0x00000000

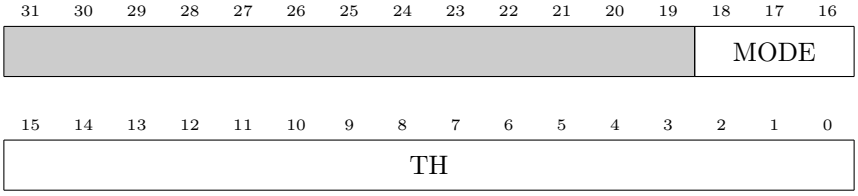


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER1 channel 1 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER1 channel 1 threshold configuration bitfield.

4.7.14 T1_TH_CHANNEL2

Address: `0x1A104054`
Reset Value: `0x00000000`

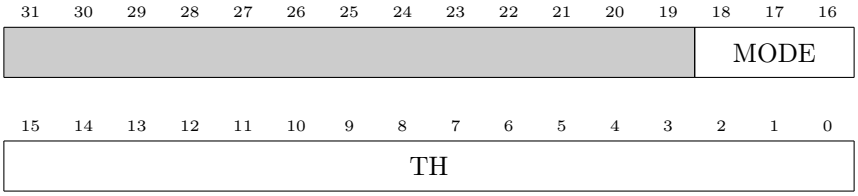


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER1 channel 2 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER1 channel 2 threshold configuration bitfield.

4.7.15 T1_TH_CHANNEL3

Address: `0x1A104058`
Reset Value: `0x00000000`

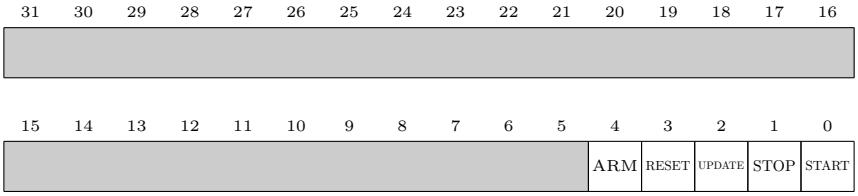


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER1 channel 3 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER1 channel 3 threshold configuration bitfield.

4.7.16 T2_CMD

Address: `0x1A104080`
Reset Value: `0x00000000`



`Bit 4` **ARM** *(R/W)* ADV_TIMER2 arm command bitfield.

`Bit 3` **RESET** *(R/W)* ADV_TIMER2 reset command bitfield.

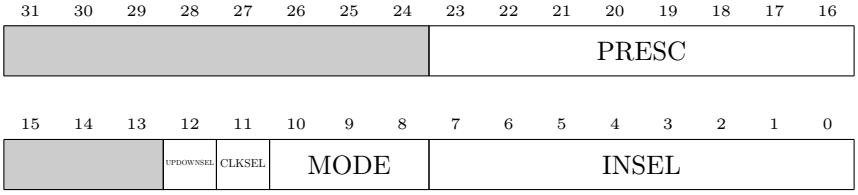
`Bit 2` **UPDATE** *(R/W)* ADV_TIMER2 update command bitfield.

`Bit 1` **STOP** *(R/W)* ADV_TIMER2 stop command bitfield.

`Bit 0` **START** *(R/W)* ADV_TIMER2 start command bitfield.

4.7.17 T2_CONFIG

Address: `0x1A104084`
Reset Value: `0x00000000`



`Bit 23 - 16` **PRESC** *(R/W)* ADV_TIMER2 prescaler value configuration bitfield.

`Bit 12` **UPDOWNSEL** *(R/W)* ADV_TIMER2 center-aligned mode configuration bitfield:
- 1'b0: The counter counts up and down alternatively.
- 1'b1: The counter counts up and resets to 0 when reach threshold.

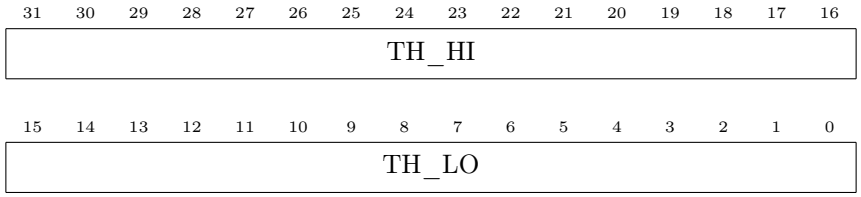
`Bit 11` **CLKSEL** *(R/W)* ADV_TIMER2 clock source configuration bitfield:
- 1'b0: FLL
- 1'b1: reference clock at 32kHz

`Bit 10 - 8` **MODE** *(R/W)* ADV_TIMER2 trigger mode configuration bitfield:
- 3'h0: trigger event at each clock cycle.
- 3'h1: trigger event if input source is 0
- 3'h2: trigger event if input source is 1
- 3'h3: trigger event on input source rising edge
- 3'h4: trigger event on input source falling edge
- 3'h5: trigger event on input source falling or rising edge
- 3'h6: trigger event on input source rising edge when armed
- 3'h7: trigger event on input source falling edge when armed

`Bit 7 - 0` **INSEL** *(R/W)* ADV_TIMER2 input source configuration bitfield:
- 0-31: GPIO[0] to GPIO[31]
- 32-35: Channel 0 to 3 of ADV_TIMER0
- 36-39: Channel 0 to 3 of ADV_TIMER1
- 40-43: Channel 0 to 3 of ADV_TIMER2
- 44-47: Channel 0 to 3 of ADV_TIMER3

4.7.18 T2_THRESHOLD

Address: `0x1A104088`
Reset Value: `0x00000000`

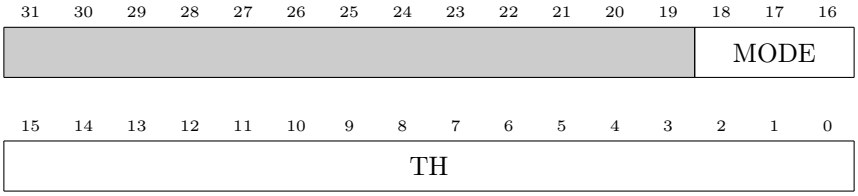


`Bit 31 - 16` **TH_HI** *(R/W)* ADV_TIMER2 threshold high part configuration bitfield. It defines end counter value.

`Bit 15 - 0` **TH_LO** *(R/W)* ADV_TIMER2 threshold low part configuration bitfield. It defines start counter value.

4.7.19 T2_TH_CHANNEL0

Address: `0x1A10408C`
Reset Value: `0x00000000`

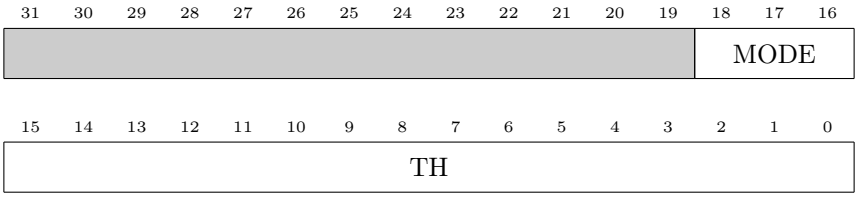


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER2 channel 0 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER2 channel 0 threshold configuration bitfield.

4.7.20 T2_TH_CHANNEL1

Address: `0x1A104090`
Reset Value: `0x00000000`

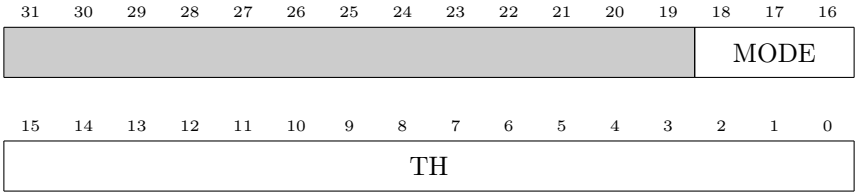


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER2 channel 1 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER2 channel 1 threshold configuration bitfield.

4.7.21 T2_TH_CHANNEL2

Address: `0x1A104094`
Reset Value: `0x00000000`

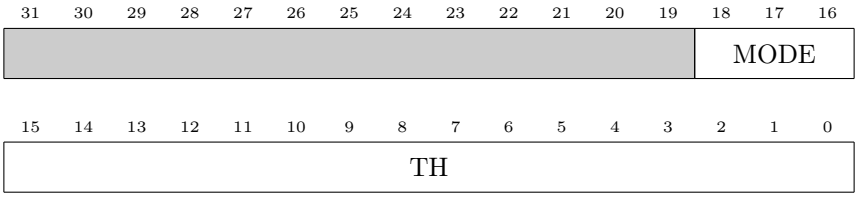


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER2 channel 2 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER2 channel 2 threshold configuration bitfield.

4.7.22 T2_TH_CHANNEL3

Address: `0x1A104098`
Reset Value: `0x00000000`

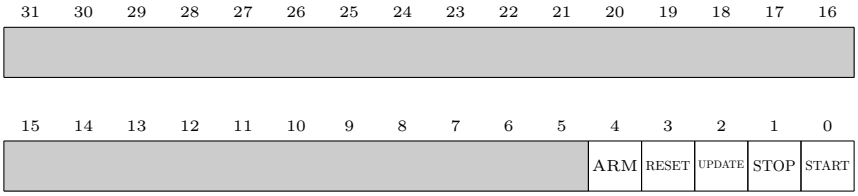


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER2 channel 3 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER2 channel 3 threshold configuration bitfield.

4.7.23 T3_CMD

Address: `0x1A1040C0`
Reset Value: `0x00000000`



`Bit 4` **ARM** *(R/W)* ADV_TIMER3 arm command bitfield.

`Bit 3` **RESET** *(R/W)* ADV_TIMER3 reset command bitfield.

`Bit 2` **UPDATE** *(R/W)* ADV_TIMER3 update command bitfield.

`Bit 1` **STOP** *(R/W)* ADV_TIMER3 stop command bitfield.

`Bit 0` **START** *(R/W)* ADV_TIMER3 start command bitfield.

4.7.24 T3_CONFIG

Address: `0x1A1040C4`
Reset Value: `0x00000000`

`Bit 23 - 16` **PRESC** *(R/W)* ADV_TIMER3 prescaler value configuration bitfield.

`Bit 12` **UPDOWNSEL** *(R/W)* ADV_TIMER3 center-aligned mode configuration bitfield:
- 1'b0: The counter counts up and down alternatively.
- 1'b1: The counter counts up and resets to 0 when reach threshold.

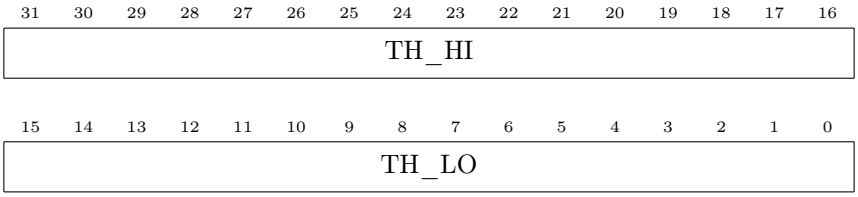
`Bit 11` **CLKSEL** *(R/W)* ADV_TIMER3 clock source configuration bitfield:
- 1'b0: FLL
- 1'b1: reference clock at 32kHz

`Bit 10 - 8` **MODE** *(R/W)* ADV_TIMER3 trigger mode configuration bitfield:
- 3'h0: trigger event at each clock cycle.
- 3'h1: trigger event if input source is 0
- 3'h2: trigger event if input source is 1
- 3'h3: trigger event on input source rising edge
- 3'h4: trigger event on input source falling edge
- 3'h5: trigger event on input source falling or rising edge
- 3'h6: trigger event on input source rising edge when armed
- 3'h7: trigger event on input source falling edge when armed

`Bit 7 - 0` **INSEL** *(R/W)* ADV_TIMER3 input source configuration bitfield:
- 0-31: GPIO[0] to GPIO[31]
- 32-35: Channel 0 to 3 of ADV_TIMER0
- 36-39: Channel 0 to 3 of ADV_TIMER1
- 40-43: Channel 0 to 3 of ADV_TIMER2
- 44-47: Channel 0 to 3 of ADV_TIMER3

4.7.25 T3_THRESHOLD

Address: 0x1A1040C8
Reset Value: 0x00000000

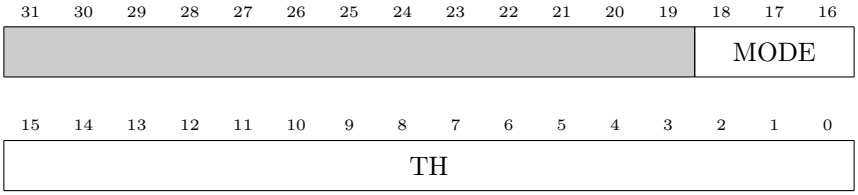


Bit 31 - 16 **TH_HI** *(R/W)* ADV_TIMER3 threshold high part configuration bitfield. It defines end counter value.

Bit 15 - 0 **TH_LO** *(R/W)* ADV_TIMER3 threshold low part configuration bitfield. It defines start counter value.

4.7.26 T3_TH_CHANNEL0

Address: 0x1A1040CC
Reset Value: 0x00000000

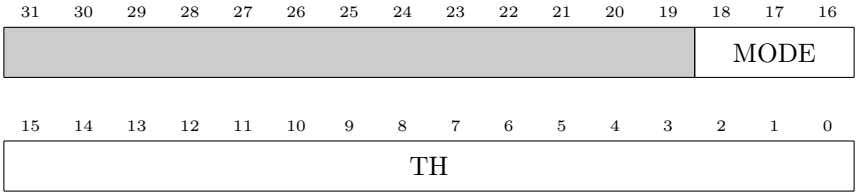


Bit 18 - 16 **MODE** *(R/W)* ADV_TIMER3 channel 0 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

Bit 15 - 0 **TH** *(R/W)* ADV_TIMER3 channel 0 threshold configuration bitfield.

4.7.27 T3_TH_CHANNEL1

Address: 0x1A1040D0
Reset Value: 0x00000000

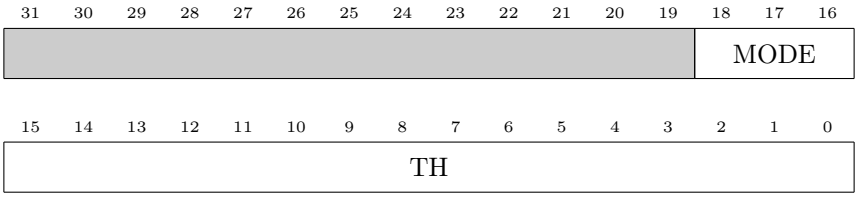


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER3 channel 1 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER3 channel 1 threshold configuration bitfield.

4.7.28 T3_TH_CHANNEL2

Address: `0x1A1040D4`
Reset Value: `0x00000000`

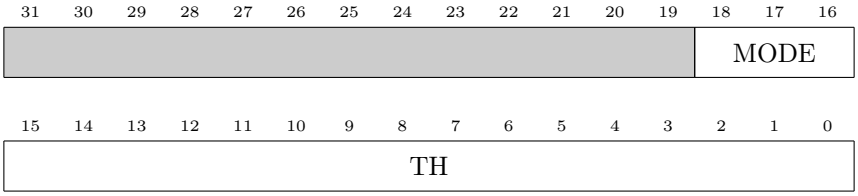


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER3 channel 2 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER3 channel 2 threshold configuration bitfield.

4.7.29 T3_TH_CHANNEL3

Address: `0x1A1040D8`
Reset Value: `0x00000000`

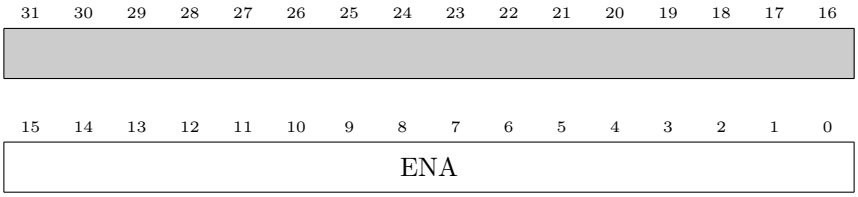


`Bit 18 - 16` **MODE** *(R/W)* ADV_TIMER3 channel 3 threshold match action on channel output signal configuration bitfield:
- 3'h0: set.
- 3'h1: toggle then next threshold match action is clear.
- 3'h2: set then next threshold match action is clear.
- 3'h3: toggle.
- 3'h4: clear.
- 3'h5: toggle then next threshold match action is set.
- 3'h6: clear then next threshold match action is set.

`Bit 15 - 0` **TH** *(R/W)* ADV_TIMER3 channel 3 threshold configuration bitfield.

4.7.30 CG

Address: `0x1A104104`
Reset Value: `0x00000000`



`Bit 15 - 0` **ENA** *(R/W)* ADV_TIMER clock gating configuration bitfield.
- ENA[i]=0: clock gate ADV_TIMERi.
- ENA[i]=1: enable ADV_TIMERi.

4.8 uDMA Subsystem

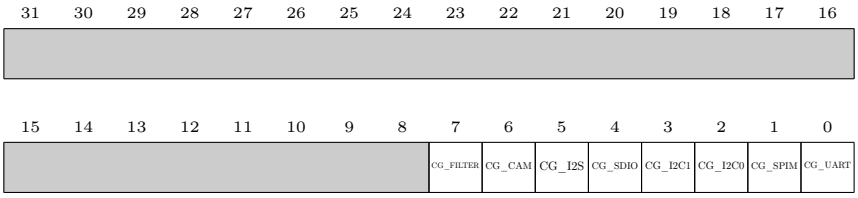
4.8.1 uDMA Control Registers

Name	Address	Size	Type	Access	Default	Description
CTRL_CFG_CG	0x1A102000	32	Config	R/W	0x00000000	uDMA peripherals clock gate configuration
CTRL_CFG_EVENT	0x1A102004	32	Config	R/W	0x00000000	uDMA peripherals external event configuration
CTRL_CFG_RST	0x1A102008	32	Config	R/W	0x00000000	uDMA peripherals reset trigger (unimplemented)

Table 4.9: uDMA Control

4.8.2 CTRL_CFG_CG

Address: 0x1A102000
Reset Value: 0x00000000



Bit 7 **CG_FILTER** *(R/W)* Defines uDMA peripherals clock gate configuration for FILTER

Bit 6 **CG_CAM** *(R/W)* Defines uDMA peripherals clock gate configuration for CAM

Bit 5 **CG_I2S** *(R/W)* Defines uDMA peripherals clock gate configuration for I2S

Bit 4 **CG_SDIO** *(R/W)* Defines uDMA peripherals clock gate configuration for SDIO

Bit 3 **CG_I2C1** *(R/W)* Defines uDMA peripherals clock gate configuration for I2C1

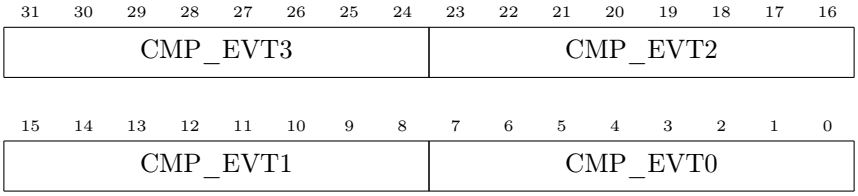
Bit 2 **CG_I2C0** *(R/W)* Defines uDMA peripherals clock gate configuration for I2C0

Bit 1 **CG_SPIM** *(R/W)* Defines uDMA peripherals clock gate configuration for SPIM

Bit 0 **CG_UART** *(R/W)* Defines uDMA peripherals clock gate configuration for UART

4.8.3 CTRL_CFG_EVENT

Address: 0x1A102004
Reset Value: 0x00000000



`Bit 31 - 24` **CMP_EVT3** *(R/W)* Forward event with ID matching CMP_EVT3 to peripherals as event3

`Bit 23 - 16` **CMP_EVT2** *(R/W)* Forward event with ID matching CMP_EVT2 to peripherals as event2

`Bit 15 - 8` **CMP_EVT1** *(R/W)* Forward event with ID matching CMP_EVT1 to peripherals as event1

`Bit 7 - 0` **CMP_EVT0** *(R/W)* Forward event with ID matching CMP_EVT0 to peripherals as event0

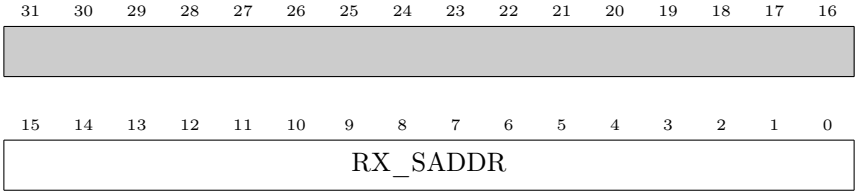
4.8.4 uDMA UART Registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A102080	32	Config	R/W	0x00000000	uDMA RX UART buffer base address configuration register.
RX_SIZE	0x1A102084	32	Config	R/W	0x00000000	uDMA RX UART buffer size configuration register.
RX_CFG	0x1A102088	32	Config	R/W	0x00000000	uDMA RX UART stream configuration register.
TX_SADDR	0x1A102090	32	Config	R/W	0x00000000	uDMA TX UART buffer base address configuration register.
TX_SIZE	0x1A102094	32	Config	R/W	0x00000000	uDMA TX UART buffer size configuration register.
TX_CFG	0x1A102098	32	Config	R/W	0x00000000	uDMA TX UART stream configuration register.
STATUS	0x1A1020A0	32	Status	R	0x00000000	uDMA UART status register.
SETUP	0x1A1020A4	32	Config	R/W	0x00000000	UDMA UART configuration register.
ERROR	0x1A1020A8	32	Status	R	0x00000000	uDMA UART Error status
IRQ_EN	0x1A1020AC	32	Config	R/W	0x00000000	uDMA UART Read or Error interrupt enable register.
VALID	0x1A1020B0	32	Status	R	0x00000000	uDMA UART Read polling data valid flag register.
DATA	0x1A1020B4	32	Data	R	0x00000000	uDMA UART Read polling data register.

Table 4.10: uDMA UART

4.8.5 RX_SADDR

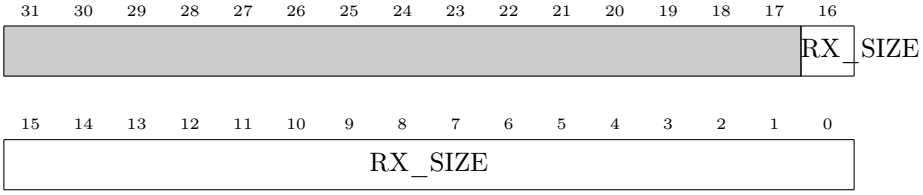
Address: `0x1A102080`
Reset Value: `0x00000000`



`Bit 15 - 0` **RX_SADDR** *(R/W)* RX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

4.8.6 RX_SIZE

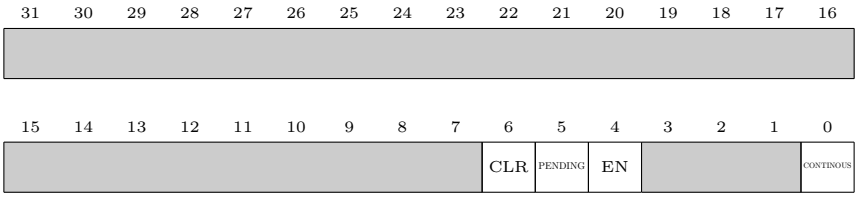
Address: `0x1A102084`
Reset Value: `0x00000000`



`Bit 16 - 0` **RX_SIZE** *(R/W)* RX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.7 RX_CFG

Address: `0x1A102088`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* RX channel clear and stop transfer:
-1'b0: disable
-1'b1: stop and clear the on-going transfer

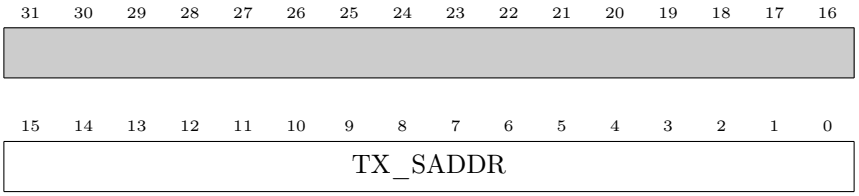
`Bit 5` **PENDING** *(R)* RX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* RX channel enable and start transfer bitfield:
-1'b0: disable
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* RX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.8 TX_SADDR

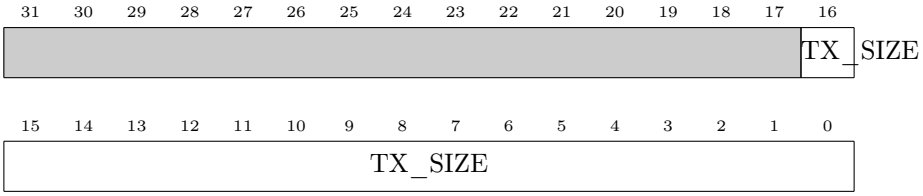
Address: `0x1A102090`
Reset Value: `0x00000000`



`Bit 15 - 0` **TX_SADDR** *(R/W)* TX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

4.8.9 TX_SIZE

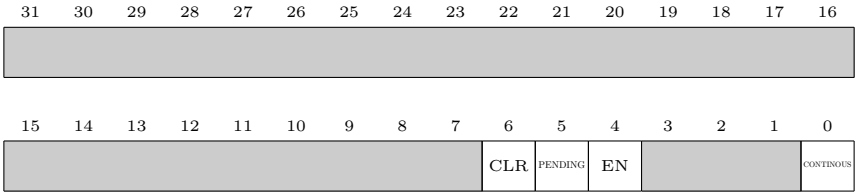
Address: `0x1A102094`
Reset Value: `0x00000000`



`Bit 16 - 0` **TX_SIZE** *(R/W)* TX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.10 TX_CFG

Address: `0x1A102098`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* TX channel clear and stop transfer bitfield:
-1'b0: disabled
-1'b1: stop and clear the on-going transfer

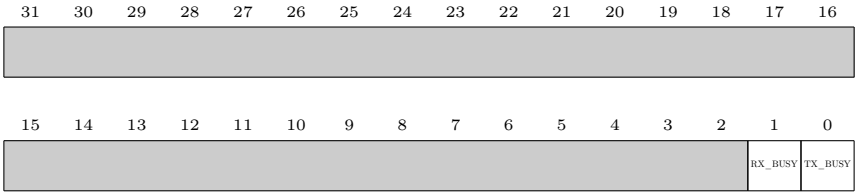
`Bit 5` **PENDING** *(R)* TX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* TX channel enable and start transfer bitfield:
-1'b0: disabled
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* TX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.11 STATUS

Address: `0x1A1020A0`
Reset Value: `0x00000000`

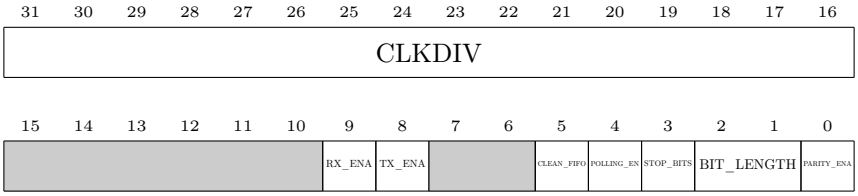


`Bit 1` **RX_BUSY** *(R)* RX busy status flag:
- 1'b0: no RX transfer on-going
- 1'b1: RX transfer on-going

`Bit 0` **TX_BUSY** *(R)* TX busy status flag:
- 1'b0: no TX transfer on-going
- 1'b1: TX transfer on-going

4.8.12 SETUP

Address: `0x1A1020A4`
Reset Value: `0x00000000`



`Bit 31 - 16` **CLKDIV** *(R/W)* UART Clock divider configuration bitfield. The baudrate is equal to SOC_FREQ/CLKDIV.

`Bit 9` **RX_ENA** *(R/W)* RX transceiver configuration bitfield:
- 1'b0: disabled
- 1'b1: enabled

`Bit 8` **TX_ENA** *(R/W)* TX transceiver configuration bitfield:
- 1'b0: disabled
- 1'b1: enabled

`Bit 5` **CLEAN_FIFO** *(R/W)* In all mode clean the RX fifo, set 1 then set 0 to realize a reset fifo:
- 1'b0: Stop Clean the RX FIFO.
- 1'b1: Clean the RX FIFO.

`Bit 4` **POLLING_EN** *(R/W)* When in uart read, use polling method to read the data, read interrupt enable flag will be ignored:
- 1'b0: Do not using polling method to read data.
- 1'b1: Using polling method to read data. Interrupt enable flag will be ignored.

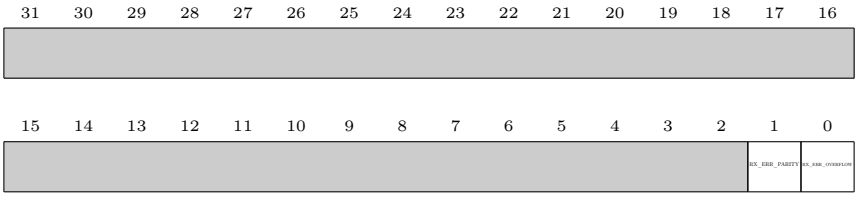
`Bit 3` **STOP_BITS** *(R/W)* Stop bits length bitfield:
- 1'b0: 1 stop bit
- 1'b1: 2 stop bits

`Bit 2 - 1` **BIT_LENGTH** *(R/W)* Character length bitfield:
- 2'b00: 5 bits
- 2'b01: 6 bits
- 2'b10: 7 bits
- 2'b11: 8 bits

`Bit 0` **PARITY_ENA** *(R/W)* Parity bit generation and check configuration bitfield:
- 1'b0: disabled
- 1'b1: enabled

4.8.13 ERROR

Address: `0x1A1020A8`
Reset Value: `0x00000000`

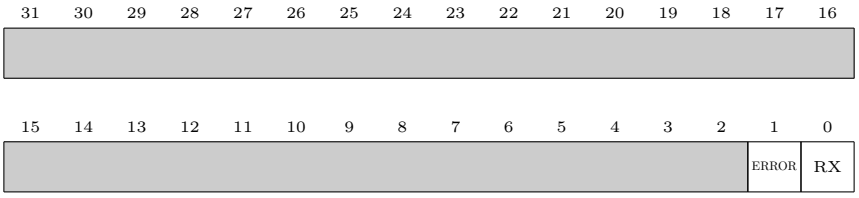


`Bit 1` **RX_ERR_PARITY** *(R)* RX parity error status flag:
- 1'b0: no error
- 1'b1: RX parity error occurred

`Bit 0` **RX_ERR_OVERFLOW** *(R)* RX overflow error status flag:
- 1'b0: no error
- 1'b1: RX overflow error occurred

4.8.14 IRQ_EN

Address: `0x1A1020AC`
Reset Value: `0x00000000`

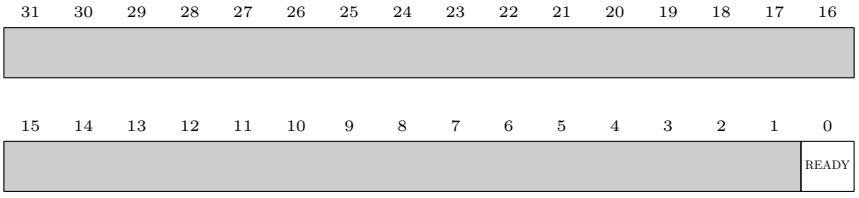


`Bit 1` **ERROR** *(R/W)* Error interrupt in enable flag:
- 1'b0: Error IRQ disable
- 1'b1: Error IRQ enable

`Bit 0` **RX** *(R/W)* Rx interrupt in enable flag:
- 1'b0: RX IRQ disable
- 1'b1: RX IRQ enable

4.8.15 VALID

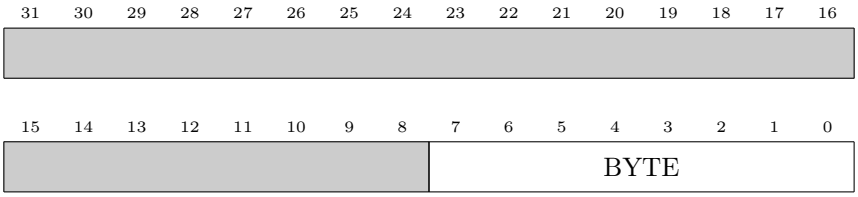
Address: `0x1A1020B0`
Reset Value: `0x00000000`



`Bit 0` **READY** *(R)* Used only in RX polling method to indicate data is ready for read:
- 1'b0: Data is not ready to read
- 1'b1: Data is ready to read

4.8.16 DATA

Address: `0x1A1020B4`
Reset Value: `0x00000000`



`Bit 7 - 0` **BYTE** *(R)* RX read data for polling or interrupt

uDMA SPI Command list

The SPI module uses a third uDMA channel (appart from the TX and RX channel) to supply it with a stream of simple configuration commands. This allows to programm complex transactions without involvement of a processor core. The processor prepares the stream of uDMA commands at an appropriate memory location. Then it configures the command chanels and the TX/RX channel accordingly. The following list contains all possible uDMA commands.

CMD Name	OP Code	CMD Description
SPI_CMD_CFG	0x0	Sets the configuration for the SPI Master IP
SPI_CMD_SOT	0x1	Sets the Chip Select (CS)
SPI_CMD_SEND_CMD	0x2	Transmits up to 16bits of data sent in the command
SPI_CMD_DUMMY	0x4	Receives a number of dummy bits (not sent to the rx interface)
SPI_CMD_WAIT	0x5	Waits an external event to move to the next instruction
SPI_CMD_TX_DATA	0x6	Sends data (max 256Kbits)
SPI_CMD_RX_DATA	0x7	Receives data (max 256Kbits)
SPI_CMD_RPT	0x8	Repeat the commands until RTP_END for N times
SPI_CMD_EOT	0x9	Clears the Chip Select (CS)
SPI_CMD_RPT_END	0xA	End of the repeat loop command
SPI_CMD_RX_CHECK	0xB	Checks up to 16 bits of data against an expected value
SPI_CMD_FULL_DUPL	0xC	Activate full duplex mode
SPI_CMD_SETUP_UCA	0xD	Sets address for uDMA tx/rx channel
SPI_CMD_SETUP_UCS	0xE	Sets size and starts uDMA tx/rx channel

SPI_CMD_CFG

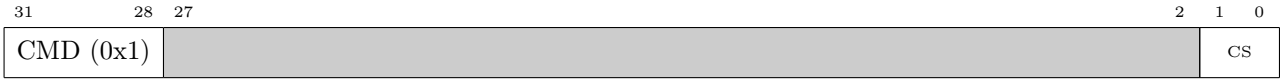
Sets the configuration for the SPI Master IP.

- **CPOL**: Sets the clock polarity:
  - 1'b0: CPOL0
  - 1'b1: CPOL1

- **CPHA**: Sets the clock phase:
  - 1'b0: CPHA0
  - 1'b1: CPHA1

SPI_CMD_SOT

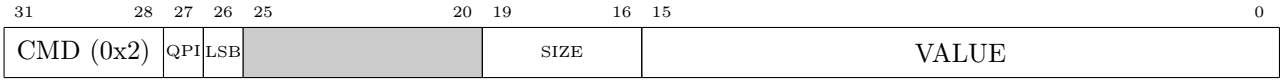
Sets the Chip Select (CS).



- **CS**: Sets the chip select (CS):
  - 2'b00: Select CSN0
  - 2'b01: Select CSN1
  - 2'b10: Select CSN2
  - 2'b11: Select CSN3

SPI_CMD_SEND_CMD

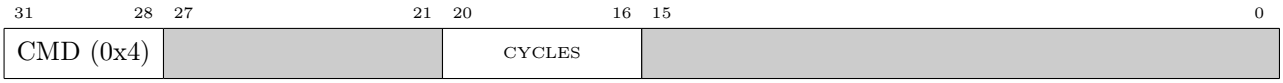
Transmits up to 16bits of data sent in the command.



- **QPI**: Sends the command using QuadSPI.
- **LSB**: Send the data starting LSB first.
- **SIZE**: Size in bits of the command ot send. The value written here is num bits-1.
- **VALUE**: The data to be send. MSB must always be at bit 15 also if SIZE is lower than 16.

SPI_CMD_DUMMY

Receives a number of dummy bits (not sent to the rx interface)



- **CYCLES**: Number of dummy cycles to perform.

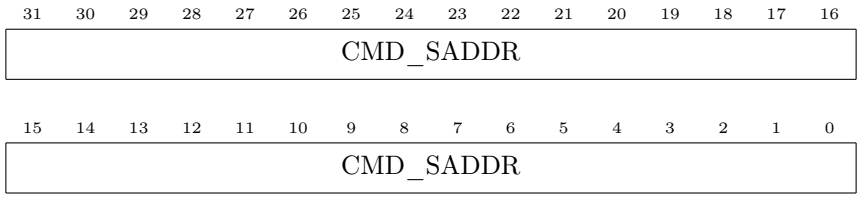
4.8.17 uDMA SPI Registers

Name	Address	Size	Type	Access	Default	Description
SPIM_RX_SADDR	0x1A102100	32	Config	R/W	0x00000000	RX SPI uDMA transfer address of associated buffer
SPIM_RX_SIZE	0x1A102104	32	Config	R/W	0x00000000	RX SPI uDMA transfer size of buffer
SPIM_RX_CFG	0x1A102108	32	Config	R/W	0x00000004	RX SPI uDMA transfer configuration
SPIM_TX_SADDR	0x1A102110	32	Config	R/W	0x00000000	TX SPI uDMA transfer address of associated buffer
SPIM_TX_SIZE	0x1A102114	32	Config	R/W	0x00000000	TX SPI uDMA transfer size of buffer
SPIM_TX_CFG	0x1A102118	32	Config	R/W	0x00000000	TX SPI uDMA transfer configuration
SPIM_CMD_SADDR	0x1A102120	32	Config	R/W	0x00000000	CMD SPI uDMA transfer address of associated buffer
SPIM_CMD_SIZE	0x1A102124	32	Config	R/W	0x00000000	CMD SPI uDMA transfer size of buffer
SPIM_CMD_CFG	0x1A102128	32	Config	R/W	0x00000004	CMD SPI uDMA transfer configuration

Table 4.12: uDMA SPI

4.8.18 SPIM_CMD_SADDR

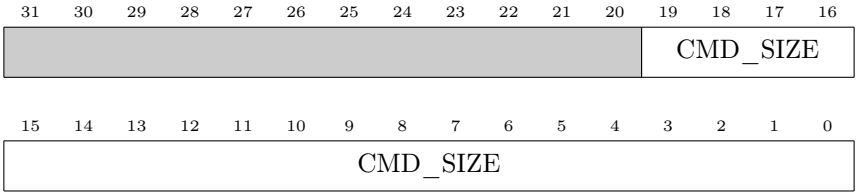
Address: 0x1A102120
Reset Value: 0x00000000



Bit 31 - 0 **CMD_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.19 SPIM_CMD_SIZE

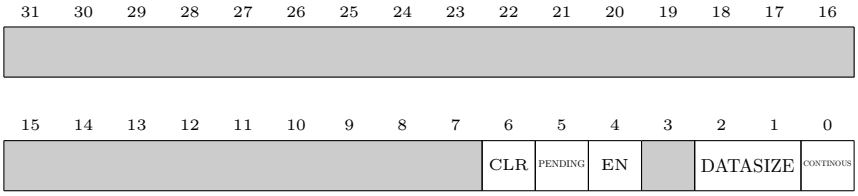
Address: 0x1A102124
Reset Value: 0x00000000



`Bit 19 - 0` **CMD_SIZE** *(R/W)* Buffer size in bytes. (1MBytes maximum)
- Read: buffer size left
- Write: set buffer size

4.8.20 SPIM_CMD_CFG

Address: `0x1A102128`
Reset Value: `0x00000004`



`Bit 6` **CLR** *(W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

`Bit 5` **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

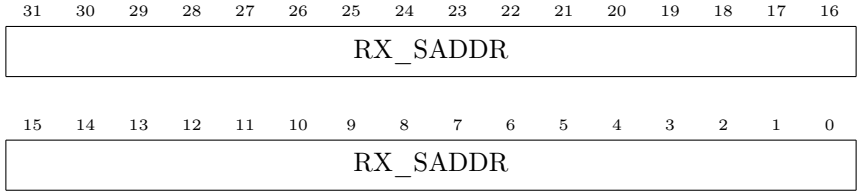
`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable
This signal is used also to queue a transfer if one is already ongoing.

`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)(default)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable
At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.21 SPIM_RX_SADDR

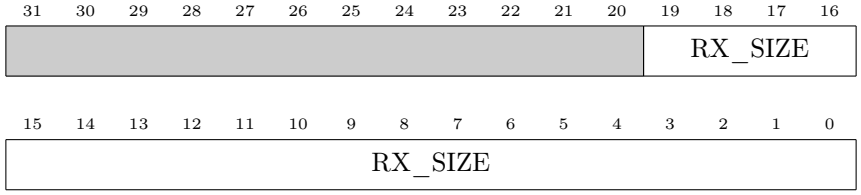
Address: `0x1A102100`
Reset Value: `0x00000000`



`Bit 31 - 0` **RX_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.22 SPIM_RX_SIZE

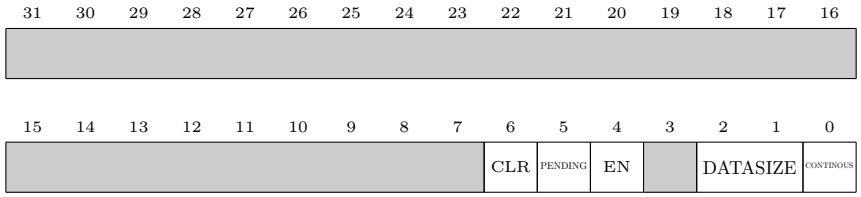
Address: `0x1A102104`
Reset Value: `0x00000000`



`Bit 19 - 0` **RX_SIZE** *(R/W)* Buffer size in bytes. (1MBytes maximum)
- Read: buffer size left
- Write: set buffer size

4.8.23 SPIM_RX_CFG

Address: `0x1A102108`
Reset Value: `0x00000004`



`Bit 6` **CLR** *(W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

`Bit 5` **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

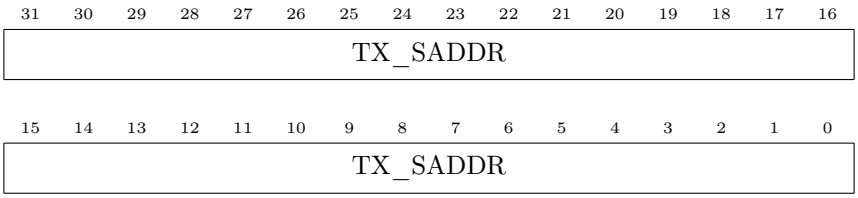
`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable
This signal is used also to queue a transfer if one is already ongoing.

`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)(default)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable
At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.24 SPIM_TX_SADDR

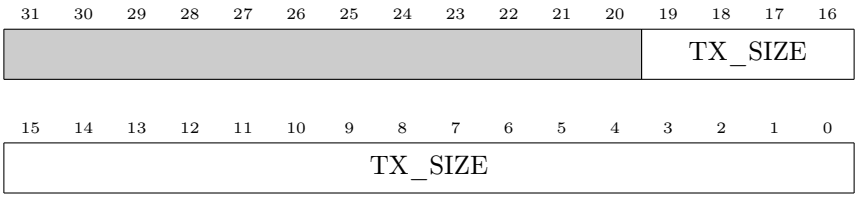
Address: `0x1A102110`
Reset Value: `0x00000000`



`Bit 31 - 0` **TX_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.25 SPIM_TX_SIZE

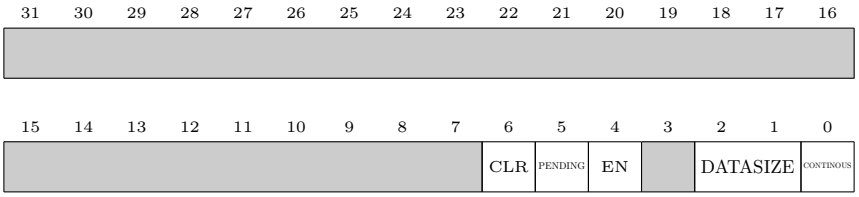
Address: `0x1A102114`
Reset Value: `0x00000000`



`Bit 19 - 0` **TX_SIZE** *(R/W)* Buffer size in bytes. (1MBytes maximum)
- Read: buffer size left
- Write: set buffer size

4.8.26 SPIM_TX_CFG

Address: `0x1A102118`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

`Bit 5` **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

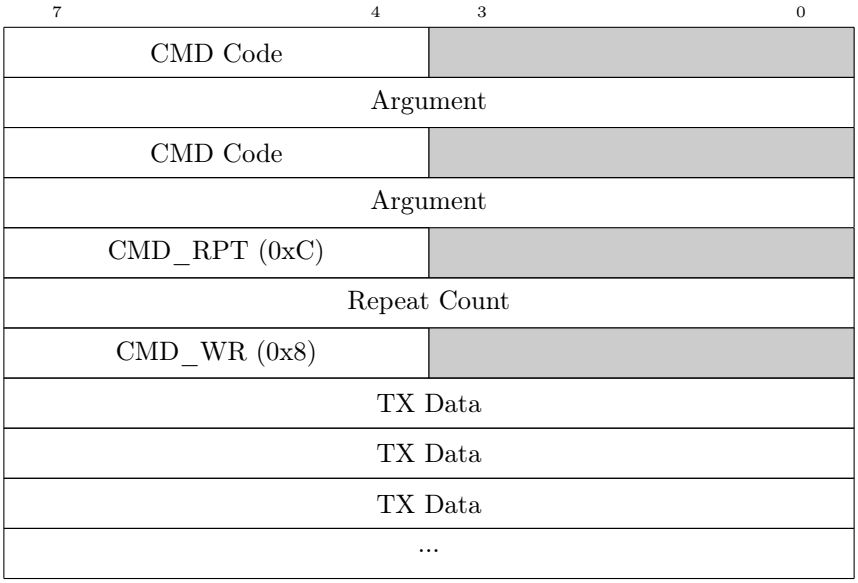
`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable
This signal is used also to queue a transfer if one is already ongoing.

`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)(default)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable
At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.27 I2C uDMA

The I2C uDMA peripheral has two uDMA channel. The RX channel is used to write the data received from an external I2C device to a configured memory location. The TX channel is used to fetch a mixture of configuration commands for the I2C peripheral and actual TX data to send to the external I2C device. The following list contains all possible uDMA commands. The arguments of these commands are supposed to follow the command byte contiguosly as follows:



I2C uDMA CMD List

CMD Name	OP Code	CMD Description
I2C_CMD_START	0x0	I2C Start of Transfer command.
I2C_CMD_STOP	0x2	I2C End of Transfer command.
I2C_CMD_RD_ACK	0x4	I2C receive data and acknowledge command.
I2C_CMD_RD_NACK	0x6	I2C receive data and not acknowledge command.
I2C_CMD_WR	0x8	I2C send data and wait acknowledge command.
I2C_CMD_WAIT	0xA	I2C wait dummy cycles command.
I2C_CMD_RPT	0xC	I2C next command repeat command.
I2C_CMD_CFG	0xE	I2C configuration command.
I2C_CMD_WAIT_EV	0x1	I2C wait uDMA external event command.

I2C_CMD_START

Command Description I2C Start of Transfer command.

Parameter *None*

I2C_CMD_STOP

Command Description I2C End of Transfer command.

Parameter *None*

I2C_CMD_RD_ACK

Command Description I2C receive data and acknowledge command.

Parameter *None*

I2C_CMD_RD_NACK

Command Description I2C receive data and not acknowledge command.

Parameter *None*

I2C_CMD_WR

Command Description I2C send data and wait acknowledge command.

Parameter The value following the I2C_CMD_WR command indicates byte value to transmit or multiple byte values to transmit if this command is preceded by a I2C_CMD_RPT command.

Parameter Size 1 byte or many bytes if preceded by a I2C_CMD_RPT command.

I2C_CMD_WAIT

Command Description I2C wait dummy cycles command.

Parameter The value following the I2C_CMD_WAIT command indicates I2C dummy clock cycles value.

Parameter Size 1 byte.

I2C_CMD_RPT

Command Description I2C next command repeat command.

Parameter The value following the I2C_CMD_RPT command indicates number of times to repeat next command.

Parameter Size 1 byte.

I2C_CMD_CFG

Command Description I2C configuration command.

Parameter The value following the I2C_CMD_CFG command indicates I2C clock divider 16bits value related to SoC clock frequency. MSB byte is sent first.

Parameter Size 2 bytes.

I2C_CMD_WAIT_EV

Command Description I2C wait uDMA external event command.

Parameter The value following the I2C_CMD_WAIT_EV command indicates selected uDMA external event ID.

Parameter Size 1 byte (bit[1:0] – event_id).

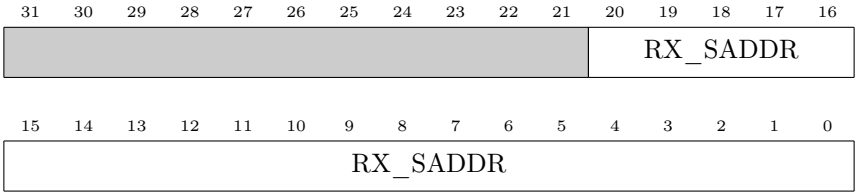
4.8.28 uDMA I2C0 Registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A102180	32	Config	R/W	0x00000000	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A102184	32	Config	R/W	0x00000000	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A102188	32	Config	R/W	0x00000000	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A102190	32	Config	R/W	0x00000000	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A102194	32	Config	R/W	0x00000000	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A102198	32	Config	R/W	0x00000000	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A1021A0	32	Config	R/W	0x00000000	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A1021A4	32	Config	R/W	0x00000000	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A1021A8	32	Config	R/W	0x00000000	uDMA CMD I2C stream configuration register.
STATUS	0x1A1021B0	32	Status	R/W	0x00000000	uDMA I2C Status register.
SETUP	0x1A1021B4	32	Config	R/W	0x00000000	uDMA I2C Configuration register.

Table 4.14: uDMA I2C0

4.8.29 RX_SADDR

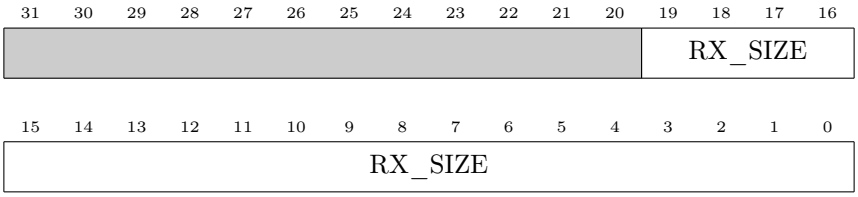
Address: 0x1A102180
Reset Value: 0x00000000



`Bit 20 - 0` **RX_SADDR** *(R/W)* RX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

4.8.30 RX_SIZE

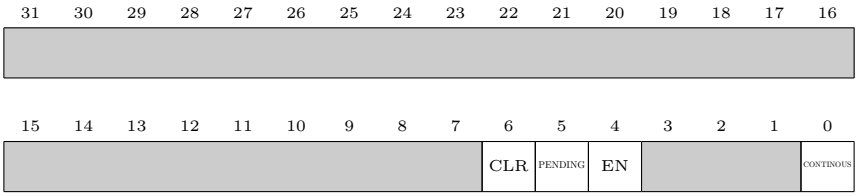
Address: `0x1A102184`
Reset Value: `0x00000000`



`Bit 19 - 0` **RX_SIZE** *(R/W)* RX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.31 RX_CFG

Address: `0x1A102188`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* RX channel clear and stop transfer:
-1'b0: disable
-1'b1: stop and clear the on-going transfer

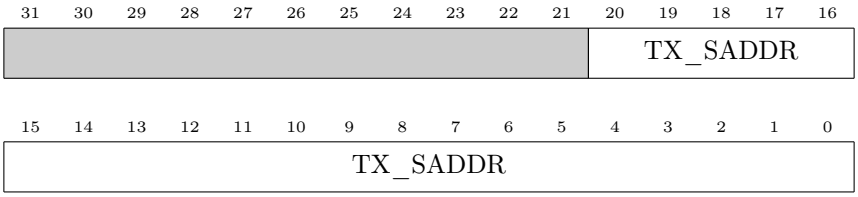
`Bit 5` **PENDING** *(R)* RX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* RX channel enable and start transfer bitfield:
-1'b0: disable
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* RX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.32 TX_SADDR

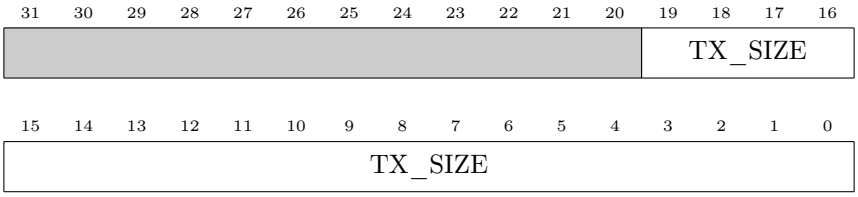
Address: `0x1A102190`
Reset Value: `0x00000000`



`Bit 20 - 0` **TX_SADDR** *(R/W)* TX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

4.8.33 TX_SIZE

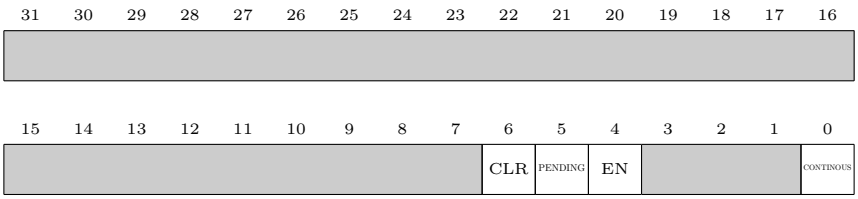
Address: `0x1A102194`
Reset Value: `0x00000000`



`Bit 19 - 0` **TX_SIZE** *(R/W)* TX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.34 TX_CFG

Address: `0x1A102198`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* TX channel clear and stop transfer bitfield:
-1'b0: disabled
-1'b1: stop and clear the on-going transfer

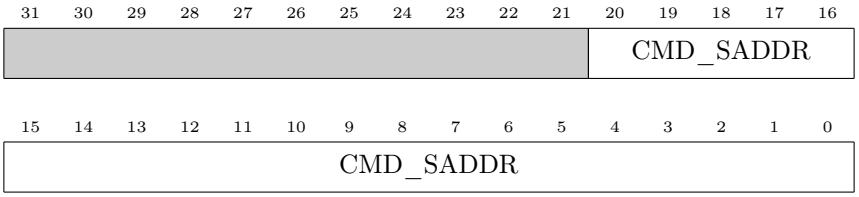
`Bit 5` **PENDING** *(R)* TX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* TX channel enable and start transfer bitfield:
-1'b0: disabled
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* TX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.35 CMD_SADDR

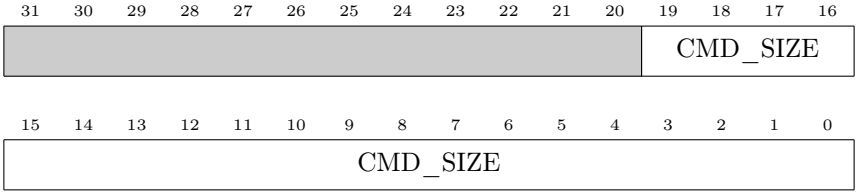
Address: `0x1A1021A0`
Reset Value: `0x00000000`



`Bit 20 - 0` **CMD_SADDR** *(R/W)* CMD buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

4.8.36 CMD_SIZE

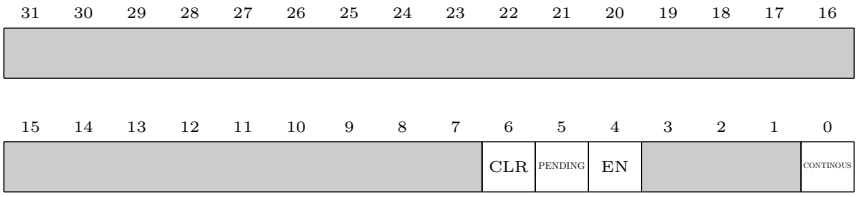
Address: `0x1A1021A4`
Reset Value: `0x00000000`



`Bit 19 - 0` **CMD_SIZE** *(R/W)* CMD buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.37 CMD_CFG

Address: `0x1A1021A8`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* CMD channel clear and stop transfer bitfield:
-1'b0: disabled
-1'b1: stop and clear the on-going transfer

`Bit 5` **PENDING** *(R)* CMD transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* CMD channel enable and start transfer bitfield:
-1'b0: disabled
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* CMD channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.38 STATUS

Address: `0x1A1021B0`
Reset Value: `0x00000000`

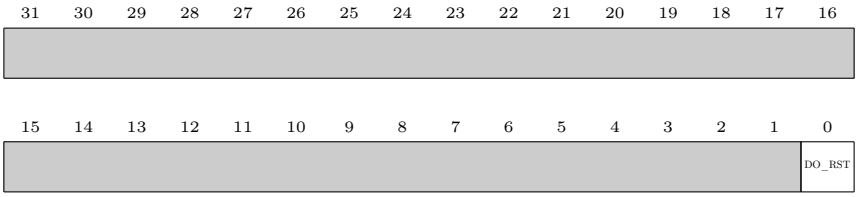
`Bit 2` **ACK** *(R)* I2C ack flag, can be polling for busy:
- 1'b0: ACK
- 1'b1: NAK

`Bit 1` **ARB_LOST** *(R/W)* I2C arbitration lost status flag:
- 1'b0: no error
- 1'b1: arbitration lost error

`Bit 0` **BUSY** *(R/W)* I2C bus busy status flag:
- 1'b0: no transfer on-going
- 1'b1: transfer on-going

4.8.39 SETUP

Address: 0x1A1021B4
Reset Value: 0x00000000



Bit 0 **DO_RST** *(R/W)* Reset command used to abort the on-going transfer and clear busy and arbitration lost status flags.

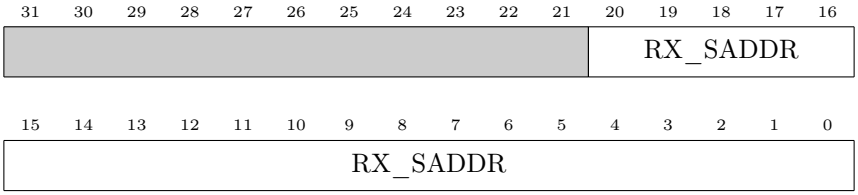
4.8.40 uDMA I2C1 Registers

Name	Address	Size	Type	Access	Default	Description
RX_SADDR	0x1A102200	32	Config	R/W	0x00000000	uDMA RX I2C buffer base address configuration register.
RX_SIZE	0x1A102204	32	Config	R/W	0x00000000	uDMA RX I2C buffer size configuration register.
RX_CFG	0x1A102208	32	Config	R/W	0x00000000	uDMA RX I2C stream configuration register.
TX_SADDR	0x1A102210	32	Config	R/W	0x00000000	uDMA TX I2C buffer base address configuration register.
TX_SIZE	0x1A102214	32	Config	R/W	0x00000000	uDMA TX I2C buffer size configuration register.
TX_CFG	0x1A102218	32	Config	R/W	0x00000000	uDMA TX I2C stream configuration register.
CMD_SADDR	0x1A102220	32	Config	R/W	0x00000000	uDMA CMD I2C buffer base address configuration register.
CMD_SIZE	0x1A102224	32	Config	R/W	0x00000000	uDMA CMD I2C buffer size configuration register.
CMD_CFG	0x1A102228	32	Config	R/W	0x00000000	uDMA CMD I2C stream configuration register.
STATUS	0x1A102230	32	Status	R/W	0x00000000	uDMA I2C Status register.
SETUP	0x1A102234	32	Config	R/W	0x00000000	uDMA I2C Configuration register.

Table 4.15: uDMA I2C1

4.8.41 RX_SADDR

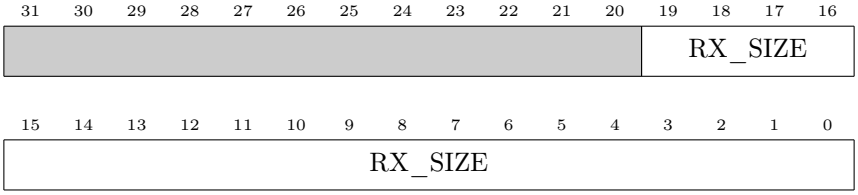
Address: 0x1A102200
Reset Value: 0x00000000



`Bit 20 - 0` **RX_SADDR** *(R/W)* RX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets RX buffer base address

4.8.42 RX_SIZE

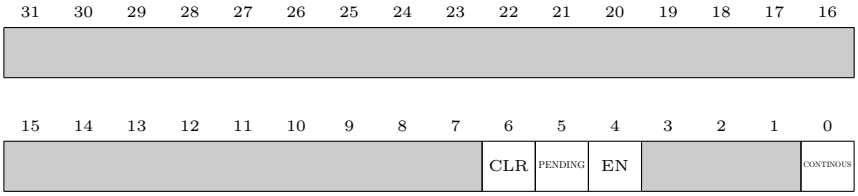
Address: `0x1A102204`
Reset Value: `0x00000000`



`Bit 19 - 0` **RX_SIZE** *(R/W)* RX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.43 RX_CFG

Address: `0x1A102208`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* RX channel clear and stop transfer:
-1'b0: disable
-1'b1: stop and clear the on-going transfer

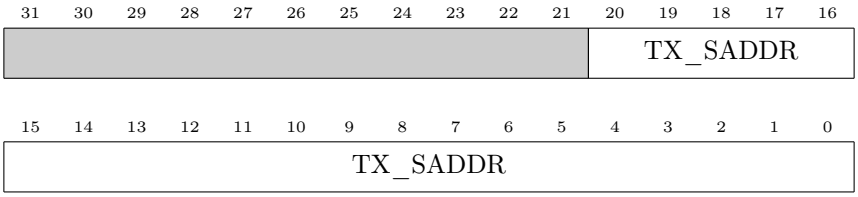
`Bit 5` **PENDING** *(R)* RX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* RX channel enable and start transfer bitfield:
-1'b0: disable
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* RX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.44 TX_SADDR

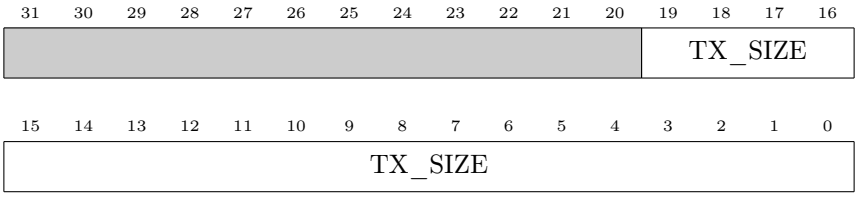
Address: `0x1A102210`
Reset Value: `0x00000000`



`Bit 20 - 0` **TX_SADDR** *(R/W)* TX buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

4.8.45 TX_SIZE

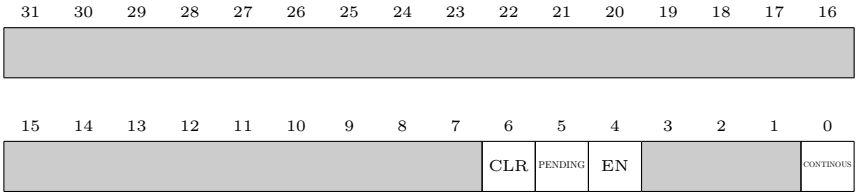
Address: `0x1A102214`
Reset Value: `0x00000000`



`Bit 19 - 0` **TX_SIZE** *(R/W)* TX buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.46 TX_CFG

Address: `0x1A102218`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* TX channel clear and stop transfer bitfield:
-1'b0: disabled
-1'b1: stop and clear the on-going transfer

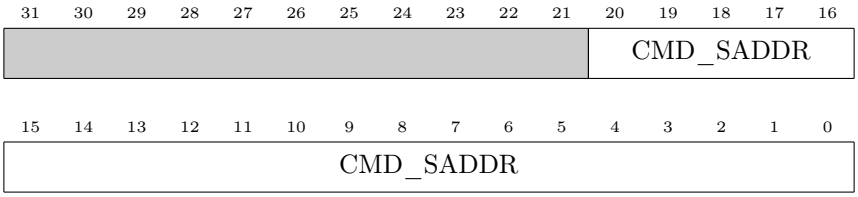
`Bit 5` **PENDING** *(R)* TX transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

`Bit 4` **EN** *(R/W)* TX channel enable and start transfer bitfield:
-1'b0: disabled
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

`Bit 0` **CONTINOUS** *(R/W)* TX channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.47 CMD_SADDR

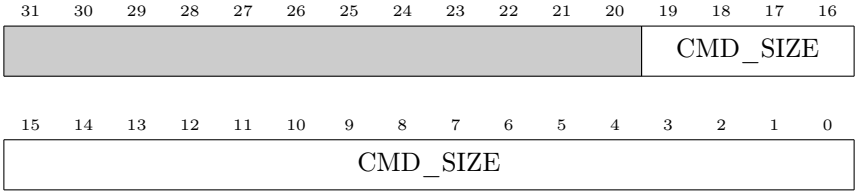
Address: `0x1A102220`
Reset Value: `0x00000000`



`Bit 20 - 0` **CMD_SADDR** *(R/W)* CMD buffer base address bitfield:
- Read: returns value of the buffer pointer until transfer is finished. Else returns 0.
- Write: sets buffer base address

4.8.48 CMD_SIZE

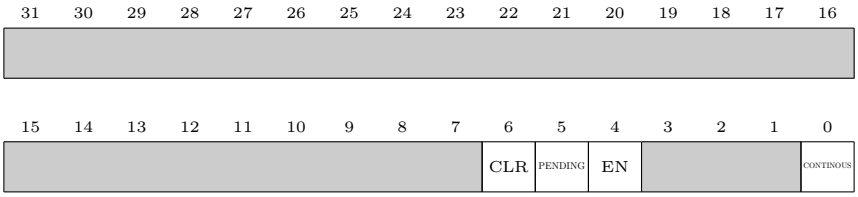
Address: `0x1A102224`
Reset Value: `0x00000000`



`Bit 19 - 0` **CMD_SIZE** *(R/W)* CMD buffer size bitfield in bytes. (128kBytes maximum)
- Read: returns remaining buffer size to transfer.
- Write: sets buffer size.

4.8.49 CMD_CFG

Address: 0x1A102228
Reset Value: 0x00000000



Bit 6 **CLR** *(W)* CMD channel clear and stop transfer bitfield:
-1'b0: disabled
-1'b1: stop and clear the on-going transfer

Bit 5 **PENDING** *(R)* CMD transfer pending in queue status flag:
-1'b0: no pending transfer in the queue
-1'b1: pending transfer in the queue

Bit 4 **EN** *(R/W)* CMD channel enable and start transfer bitfield:
-1'b0: disabled
-1'b1: enable and start the transfer
This signal is used also to queue a transfer if one is already ongoing.

Bit 0 **CONTINOUS** *(R/W)* CMD channel continuous mode bitfield:
-1'b0: disabled
-1'b1: enabled
At the end of the buffer transfer, the uDMA reloads the address / buffer size and starts a new transfer.

4.8.50 STATUS

Address: 0x1A102230
Reset Value: 0x00000000

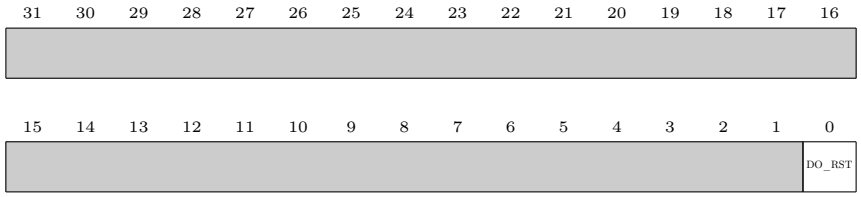
Bit 2 **ACK** *(R)* I2C ack flag, can be polling for busy:
- 1'b0: ACK
- 1'b1: NAK

Bit 1 **ARB_LOST** *(R/W)* I2C arbitration lost status flag:
- 1'b0: no error
- 1'b1: arbitration lost error

Bit 0 **BUSY** *(R/W)* I2C bus busy status flag:
- 1'b0: no transfer on-going
- 1'b1: transfer on-going

4.8.51 SETUP

Address: 0x1A102234
Reset Value: 0x00000000



Bit 0 **DO_RST** *(R/W)* Reset command used to abort the on-going transfer and clear busy and arbitration lost status flags.

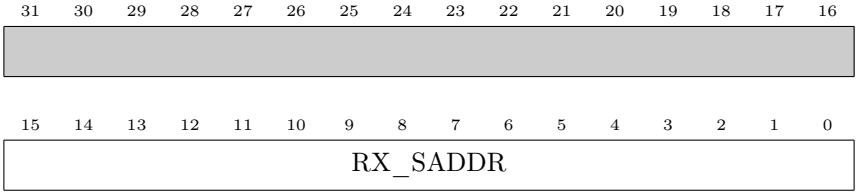
4.8.52 uDMA I2S Registers

Name	Address	Size	Type	Access	Default	Description
I2S_RX_SADDR	0x1A102300	32	Config	R/W	0x00000000	RX Channel 0 I2S uDMA transfer address of associated buffer
I2S_RX_SIZE	0x1A102304	32	Config	R/W	0x00000000	RX Channel 0 I2S uDMA transfer size of buffer
I2S_RX_CFG	0x1A102308	32	Config	R/W	0x00000004	RX Channel 0 I2S uDMA transfer configuration
I2S_TX_SADDR	0x1A102310	32	Config	R/W	0x00000000	TX Channel I2S uDMA transfer address of associated buffer
I2S_TX_SIZE	0x1A102314	32	Config	R/W	0x00000000	TX Channel I2S uDMA transfer size of buffer
I2S_TX_CFG	0x1A102318	32	Config	R/W	0x00000004	TX Channel I2S uDMA transfer configuration
I2S_CLKCFG_SETUP	0x1A102320	32	Config	R/W	0x00000000	Clock configuration for both master, slave and pdm
I2S_SLV_SETUP	0x1A102324	32	Config	R/W	0x00000000	Configuration of I2S slave
I2S_MST_SETUP	0x1A102328	32	Config	R/W	0x00000000	Configuration of I2S master
I2S_PDM_SETUP	0x1A10232C	32	Config	R/W	0x00000000	Configuration of PDM module

Table 4.16: uDMA I2S

4.8.53 I2S_RX_SADDR

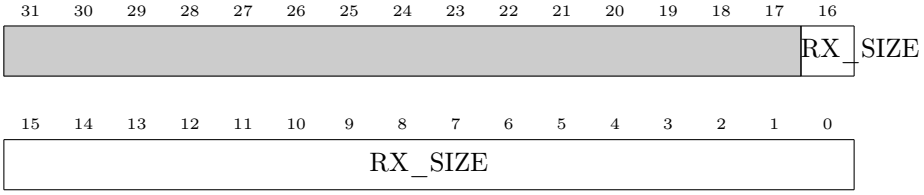
Address: 0x1A102300
Reset Value: 0x00000000



Bit 15 - 0 **RX_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.54 I2S_RX_SIZE

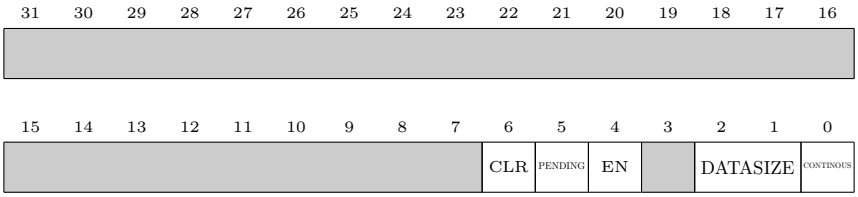
Address: 0x1A102304
Reset Value: 0x00000000



Bit 16 - 0 **RX_SIZE** *(R/W)* Buffer size in byte. (128kBytes maximum)
- Read: buffer size left
- Write: set buffer size

4.8.55 I2S_RX_CFG

Address: 0x1A102308
Reset Value: 0x00000004



Bit 6 **CLR** *(W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

Bit 5 **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

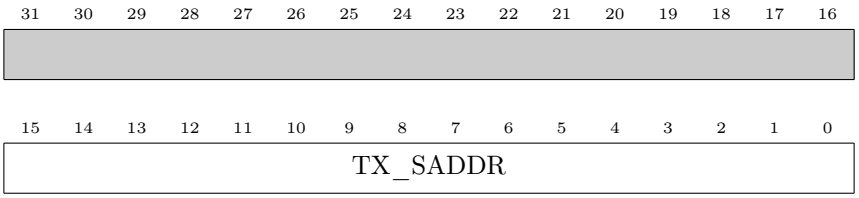
`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable
This signal is used also to queue a transfer if one is already ongoing.

`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable
At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.56 I2S_TX_SADDR

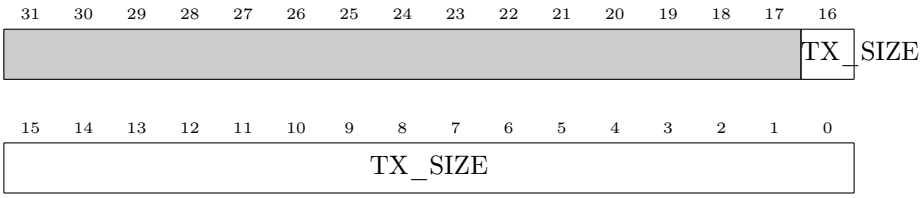
Address: `0x1A102310`
Reset Value: `0x00000000`



`Bit 15 - 0` **TX_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.57 I2S_TX_SIZE

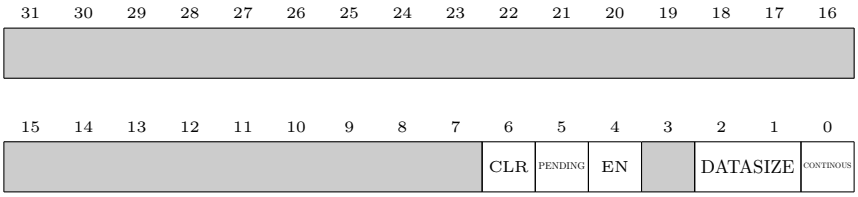
Address: `0x1A102314`
Reset Value: `0x00000000`



`Bit 16 - 0` **TX_SIZE** *(R/W)* Buffer size in byte. (128kBytes maximum)
- Read: buffer size left
- Write: set buffer size

4.8.58 I2S_TX_CFG

Address: `0x1A102318`
Reset Value: `0x00000004`



`Bit 6` **CLR** *(R/W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

`Bit 5` **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

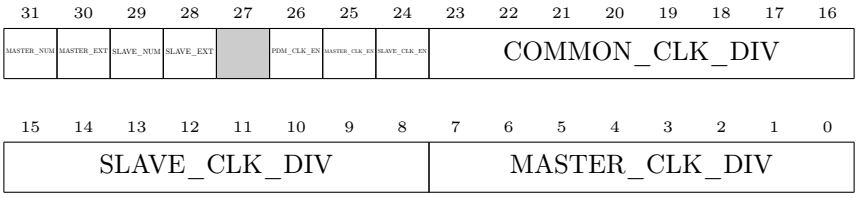
`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable
This signal is used also to queue a transfer if one is already ongoing.

`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable
At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.59 I2S_CLKCFG_SETUP

Address: `0x1A102320`
Reset Value: `0x00000000`



`Bit 31` **MASTER_NUM** *(R/W)* Selects master clock source(either ext or generated):
-1'b0:selects master
-1'b1:selects slave

`Bit 30` **MASTER_EXT** *(R/W)* When set uses external clock for master

`Bit 29` **SLAVE_NUM** *(R/W)* Selects slave clock source(either ext or generated):
-1'b0:selects master
-1'b1:selects slave

`Bit 28` **SLAVE_EXT** *(R/W)* When set uses external clock for slave

`Bit 26` **PDM_CLK_EN** *(R/W)* When enabled slave output clock is taken from PDM module

`Bit 25` **MASTER_CLK_EN** *(R/W)* Enables Master clock

`Bit 24` **SLAVE_CLK_EN** *(R/W)* Enables Slave clock

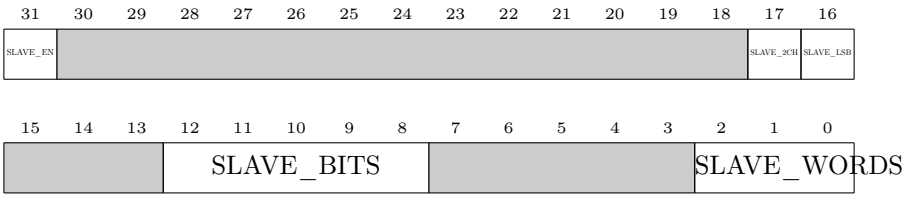
`Bit 23 - 16` **COMMON_CLK_DIV** *(R/W)* MSBs of both master and slave clock divider

`Bit 15 - 8` **SLAVE_CLK_DIV** *(R/W)* LSB of slave clock divider

`Bit 7 - 0` **MASTER_CLK_DIV** *(R/W)* LSB of master clock divider

4.8.60 I2S_SLV_SETUP

Address: `0x1A102324`
Reset Value: `0x00000000`



`Bit 31` **SLAVE_EN** *(R/W)* Enables the Slave

`Bit 17` **SLAVE_2CH** *(R/W)* Enables both channels

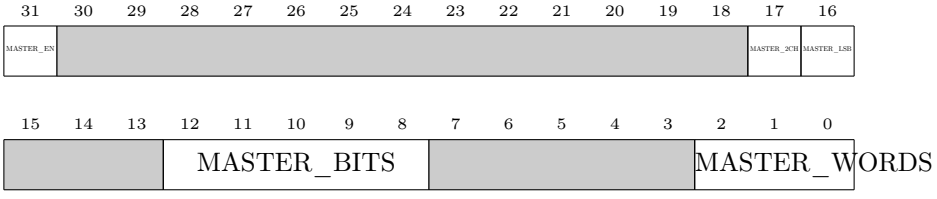
`Bit 16` **SLAVE_LSB** *(R/W)* Enables LSB shifting

`Bit 12 - 8` **SLAVE_BITS** *(R/W)* Sets how many bits per word

`Bit 2 - 0` **SLAVE_WORDS** *(R/W)* Sets how many words for each I2S phase

4.8.61 I2S_MST_SETUP

Address: `0x1A102328`
Reset Value: `0x00000000`



`Bit 31` **MASTER_EN** *(R/W)* Enables the Master

`Bit 17` **MASTER_2CH** *(R/W)* Enables both channels

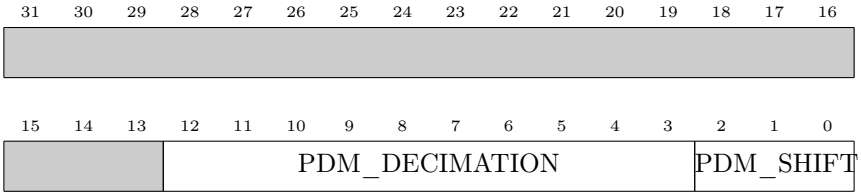
Bit 16 **MASTER_LSB** *(R/W)* Enables LSB shifting

Bit 12 - 8 **MASTER_BITS** *(R/W)* Sets how many bits per word

Bit 2 - 0 **MASTER_WORDS** *(R/W)* Sets how many words for each I2S phase

4.8.62 I2S_PDM_SETUP

Address: 0x1A10232C
Reset Value: 0x00000000



Bit 12 - 3 **PDM_DECIMATION** *(R/W)* Sets the decimation ratio of the filter

Bit 2 - 0 **PDM_SHIFT** *(R/W)* Shifts the output of the filter

4.8.63 uDMA Camera Interface Registers

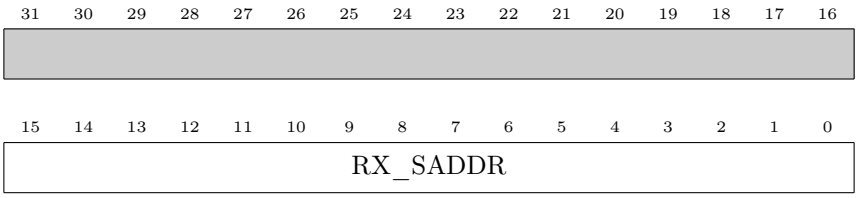
Name	Address	Size	Type	Access	Default	Description
CAM_RX_SADDR	0x1A102380	32	Config	R/W	0x00000000	RX Camera uDMA transfer address of associated buffer register
CAM_RX_SIZE	0x1A102384	32	Config	R/W	0x00000000	RX Camera uDMA transfer size of buffer register
CAM_RX_CFG	0x1A102388	32	Config	R/W	0x00000000	RX Camera uDMA transfer configuration register
CAM_CFG_GLOB	0x1A1023A0	32	Config	R/W	0x00000000	Global configuration register
CAM_CFG_LL	0x1A1023A4	32	Config	R/W	0x00000000	Lower Left corner configuration register

CAM_CFG_UR	0x1A1023A8	32	Config	R/W	0x00000000	Upper Right corner configuration register
CAM_CFG_SIZE	0x1A1023AC	32	Config	R/W	0x00000000	Horizontal Resolution configuration register
CAM_CFG_FILTER	0x1A1023B0	32	Config	R/W	0x00000000	RGB coefficients configuration register
CAM_VSYNC_POLARITY	0x1A1023B4	32	Config	R/W	0x00000000	VSYNC Polarity register

Table 4.17: uDMA Camera Interface

4.8.64 CAM_RX_SADDR

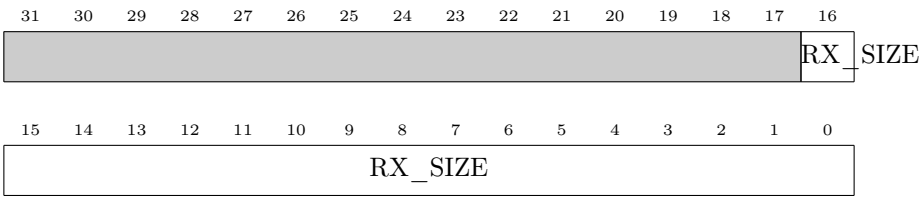
Address: 0x1A102380
Reset Value: 0x00000000



Bit 15 - 0 **RX_SADDR** *(R/W)* Configure pointer to memory buffer:
- Read: value of the pointer until transfer is over. Else returns 0
- Write: set Address Pointer to memory buffer start address

4.8.65 CAM_RX_SIZE

Address: 0x1A102384
Reset Value: 0x00000000

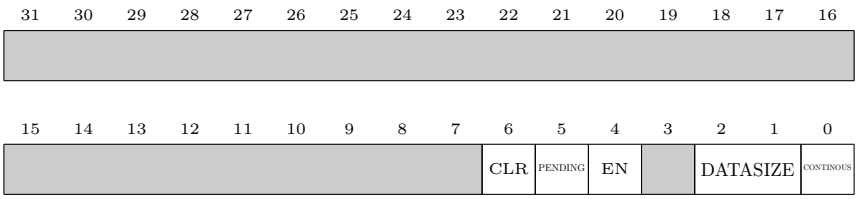


`Bit 16 - 0` **RX_SIZE** *(R/W)* Buffer size in bytes. (128kBytes maximum)
- Read: buffer size left
- Write: set buffer size

NOTE: Careful with size in byte. If you use uncompressed pixel data mapped on 16 bits, you have to declare buffer size in bytes even if buffer type is short.

4.8.66 CAM_RX_CFG

Address: `0x1A102388`
Reset Value: `0x00000000`



`Bit 6` **CLR** *(W)* Channel clear and stop transfer:
-1'b0: disable
-1'b1: enable

`Bit 5` **PENDING** *(R)* Transfer pending in queue status flag:
-1'b0: free
-1'b1: pending

`Bit 4` **EN** *(R/W)* Channel enable and start transfer:
-1'b0: disable
-1'b1: enable

This signal is used also to queue a transfer if one is already ongoing.

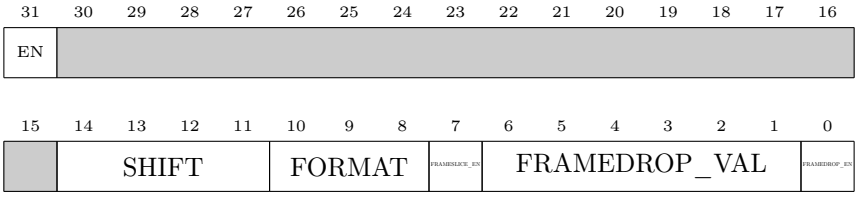
`Bit 2 - 1` **DATASIZE** *(R/W)* Channel transfer size used to increment uDMA buffer address pointer:
- 2'b00: +1 (8 bits)
- 2'b01: +2 (16 bits)
- 2'b10: +4 (32 bits)
- 2'b11: +0

`Bit 0` **CONTINOUS** *(R/W)* Channel continuous mode:
-1'b0: disable
-1'b1: enable

At the end of the buffer the uDMA reloads the address and size and starts a new transfer.

4.8.67 CAM_CFG_GLOB

Address: `0x1A1023A0`
Reset Value: `0x00000000`



`Bit 31` **EN** *(R/W)* Enable data rx from camera interface.
The enable/disable happens only at the start of a frame.
- 1'b0: disable
- 1'b1: enable

`Bit 14 - 11` **SHIFT** *(R/W)* Right shift of final pixel value (DivFactor)
NOTE: not used if FORMAT == BYPASS

`Bit 10 - 8` **FORMAT** *(R/W)* Input frame format:
- 3'b000: RGB565
- 3'b001: RGB555
- 3'b010: RGB444
- 3'b100: BYPASS_LITEND
- 3'b101: BYPASS_BIGEND

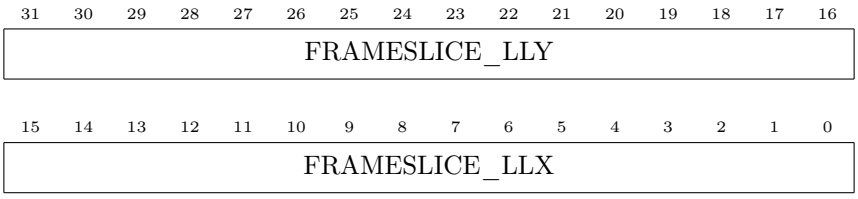
`Bit 7` **FRAMESLICE_EN** *(R/W)* Input frame slicing:
- 1'b0: disable
- 1'b1: enable

`Bit 6 - 1` **FRAMEDROP_VAL** *(R/W)* Sets how many frames should be dropped after each received.

`Bit 0` **FRAMEDROP_EN** *(R/W)* Frame dropping:
- 1'b0: disable
- 1'b1: enable

4.8.68 CAM_CFG_LL

Address: `0x1A1023A4`
Reset Value: `0x00000000`

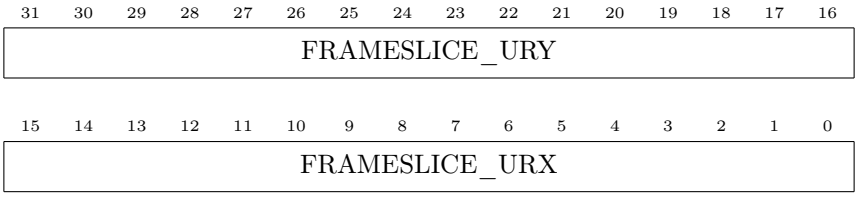


`Bit 31 - 16` **FRAMESLICE_LLY** *(R/W)* Y coordinate of lower left corner of slice

`Bit 15 - 0` **FRAMESLICE_LLX** *(R/W)* X coordinate of lower left corner of slice

4.8.69 CAM_CFG_UR

Address: `0x1A1023A8`
Reset Value: `0x00000000`

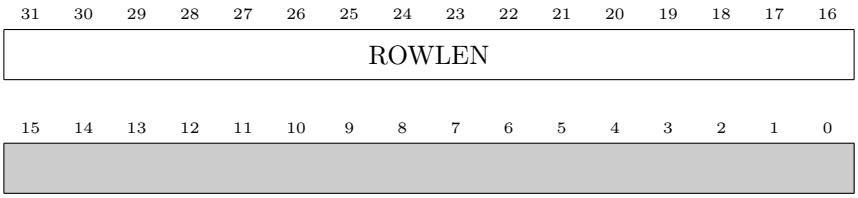


`Bit 31 - 16` **FRAMESLICE_URY** *(R/W)* Y coordinate of upper right corner of slice

`Bit 15 - 0` **FRAMESLICE_URX** *(R/W)* X coordinate of upper right corner of slice

4.8.70 CAM_CFG_SIZE

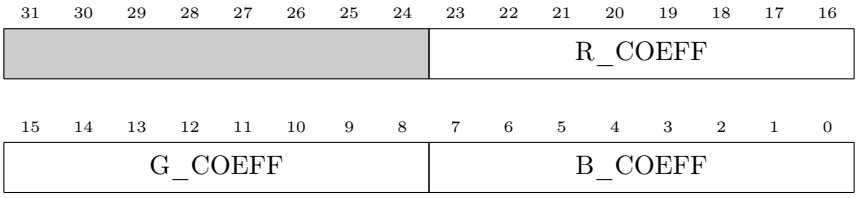
Address: `0x1A1023AC`
Reset Value: `0x00000000`



`Bit 31 - 16` **ROWLEN** *(R/W)* Horizontal Resolution. It is used for slice mode. Value set into the bitfield must be equal to (rowlen-1).

4.8.71 CAM_CFG_FILTER

Address: `0x1A1023B0`
Reset Value: `0x00000000`



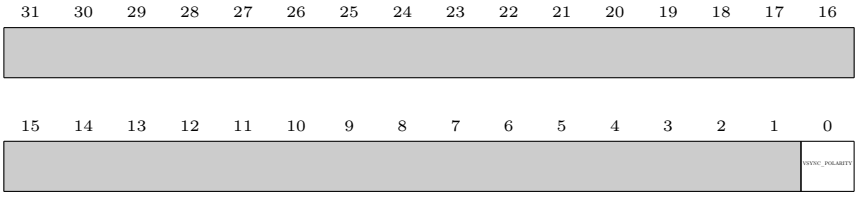
`Bit 23 - 16` **R_COEFF** *(R/W)* Coefficient that multiplies the R component
NOTE: not used if FORMAT == BYPASS

`Bit 15 - 8` **G_COEFF** *(R/W)* Coefficient that multiplies the G component
NOTE: not used if FORMAT == BYPASS

`Bit 7 - 0` **B_COEFF** *(R/W)* Coefficient that multiplies the B component
NOTE: not used if FORMAT == BYPASS

4.8.72 CAM_VSYNC_POLARITY

Address: `0x1A1023B4`
Reset Value: `0x00000000`



`Bit 0` **VSYNC_POLARITY** *(R/W)* Set vsync polarity of CPI.
- 1'b0: Active 0
- 1'b1: Active 1

4.8.73 uDMA Filter Registers

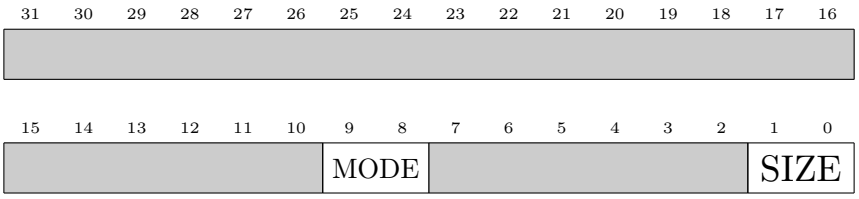
Name	Address	Size	Type	Access	Default	Description
REG_TX_CH0_ADD	0x1A102400	32	Config	R/W	0x00000000	FILTER tx channel 0 address register
REG_TX_CH0_CFG	0x1A102404	32	Config	R/W	0x00000000	FILTER tx channel 0 configuration register
REG_TX_CH0_LEN0	0x1A102408	32	Config	R/W	0x00000000	FILTER tx channel 0 length0 register
REG_TX_CH0_LEN1	0x1A10240C	32	Config	R/W	0x00000000	FILTER tx channel 0 length1 register
REG_TX_CH0_LEN2	0x1A102410	32	Config	R/W	0x00000000	FILTER tx channel 0 length2 register
REG_TX_CH1_ADD	0x1A102414	32	Config	R/W	0x00000000	FILTER tx channel 1 address register
REG_TX_CH1_CFG	0x1A102418	32	Config	R/W	0x00000000	FILTER tx channel 1 configuration register
REG_TX_CH1_LEN0	0x1A10241C	32	Config	R/W	0x00000000	FILTER tx channel 1 length0 register
REG_TX_CH1_LEN1	0x1A102420	32	Config	R/W	0x00000000	FILTER tx channel 1 length1 register
REG_TX_CH1_LEN2	0x1A102424	32	Config	R/W	0x00000000	FILTER tx channel 1 length2 register
REG_RX_CH_ADD	0x1A102428	32	Config	R/W	0x00000000	FILTER RX channel address register
REG_RX_CH_CFG	0x1A10242C	32	Config	R/W	0x00000000	FILTER RX channel configuration register
REG_RX_CH_LEN0	0x1A102430	32	Config	R/W	0x00000000	FILTER RX channel length0 register
REG_RX_CH_LEN1	0x1A102434	32	Config	R/W	0x00000000	FILTER RX channel length1 register
REG_RX_CH_LEN2	0x1A102438	32	Config	R/W	0x00000000	FILTER RX channel length2 register

REG_AU_CFG	0x1A10243C	32	Config	R/W	0x00000000	FILTER arithmetic unit configuration register
REG_AU_REG0	0x1A102440	32	Config	R/W	0x00000000	FILTER arithmetic unit 0 register
REG_AU_REG1	0x1A102444	32	Config	R/W	0x00000000	FILTER arithmetic unit 1 register
REG_BINCU_TH	0x1A102448	32	Config	R/W	0x00000000	FILTER binarization threshold register
REG_BINCU_CNT	0x1A10244C	32	Config	R/W	0x00000000	FILTER binarization count register
REG_BINCU_SETUP	0x1A102450	32	Config	R/W	0x00000000	FILTER binarization datasize format register
REG_BINCU_VAL	0x1A102454	32	Status	R	0x00000000	FILTER binarization result count register
REG_FILT	0x1A102458	32	Config	R/W	0x00000000	FILTER control mode register
REG_FILT_CMD	0x1A10245C	32	Config	R/W	0x00000000	FILTER start register
REG_STATUS	0x1A102460	32	Status	R/W	0x00000000	FILTER status register

Table 4.18: uDMA Filter

4.8.74 REG_TX_CH0_CFG

Address: 0x1A102404
Reset Value: 0x00000000

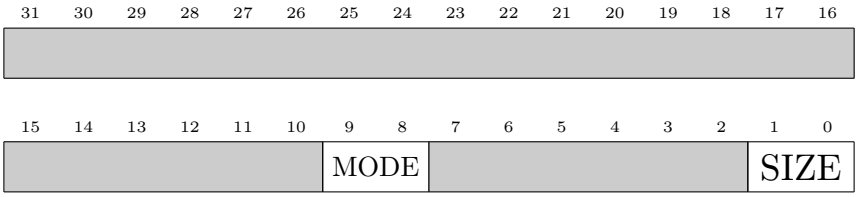


Bit 9 - 8 **MODE** *(R)* Data transfer mode:
- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

Bit 1 - 0 **SIZE** *(R)* Data transfer format:
- 2'b00: 8-bit
- 2'b01:16-bit
- 2;b10:32-bit

4.8.75 REG_TX_CH1_CFG

Address: `0x1A102418`
Reset Value: `0x00000000`

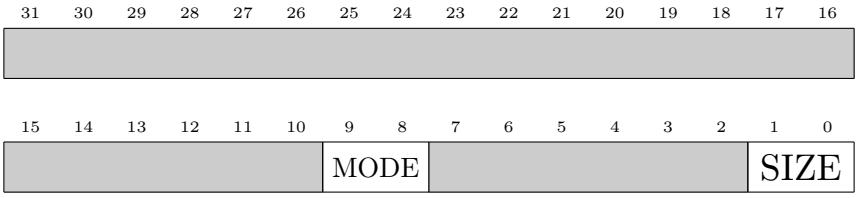


`Bit 9 - 8` **MODE** *(R/W)* Data transfer mode:
- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

`Bit 1 - 0` **SIZE** *(R)* Data transfer format:
- 2'b00: 8-bit
- 2'b01:16-bit
- 2'b10:32-bit

4.8.76 REG_RX_CH_CFG

Address: `0x1A10242C`
Reset Value: `0x00000000`

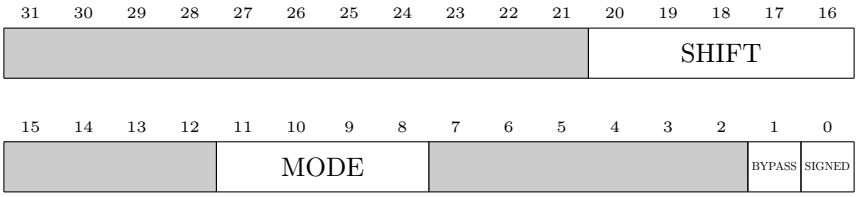


`Bit 9 - 8` **MODE** *(R/W)* Data transfer mode:
- 2'b00: Linear
- 2'b01: Sliding
- 2;b10:Circular
- 2;b11:2D

`Bit 1 - 0` **SIZE** *(R/W)* Data transfer format:
- 2'b00: 8-bit
- 2'b01:16-bit
- 2;b10:32-bit

4.8.77 REG_AU_CFG

Address: 0x1A10243C
Reset Value: 0x00000000



Bit 20 - 16 **SHIFT** *(R/W)* Arithmetic Unit shift window size, (0 – 31).

Bit 11 - 8 **MODE** *(R/W)* Arithmetic Unit mode:
-4'b0000: AU_MODE_AxB
-4'b0001: AU_MODE_AxB+REG0
-4'b0010: AU_MODE_AxB accumulation
-4'b0011: AU_MODE_AxA
-4'b0100: AU_MODE_AxA+B
-4'b0101: AU_MODE_AxA-B
-4'b0110: AU_MODE_AxA accumulation
-4'b0111: AU_MODE_AxA+REG0
-4'b1000: AU_MODE_AxREG1
-4'b1001: AU_MODE_AxREG1+B
-4'b1010: AU_MODE_AxREG1-B
-4'b1011: AU_MODE_AxREG1+REG0
-4'b1100: AU_MODE_AxREG1 accumulation
-4'b1101: AU_MODE_A+B
-4'b1110: AU_MODE_A-B
-4'b1111: AU_MODE_A+REG0

Bit 1 **BYPASS** *(R/W)* Arithmetic Unit bypass or not.
-1'b0: not bypass AU
-1'b1: bypass AU

Bit 0 **SIGNED** *(R/W)* Arithmetic Unit result signed or not.
-1'b0: not signed
-1'b1: signed

4.8.78 REG_BINCU_CNT

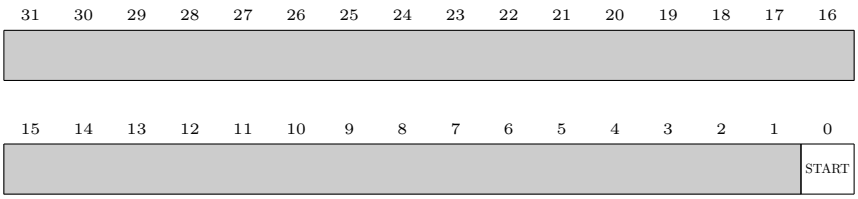
Address: 0x1A10244C
Reset Value: 0x00000000

`Bit 31` **EN** *(R/W)* Binarization and counting unit enable:
-1'b0: not enable
-1'b1: enable

`Bit 19 - 0` **COUNT** *(R/W)* Binarization and counting unit count value set.

4.8.79 REG_FILT_CMD

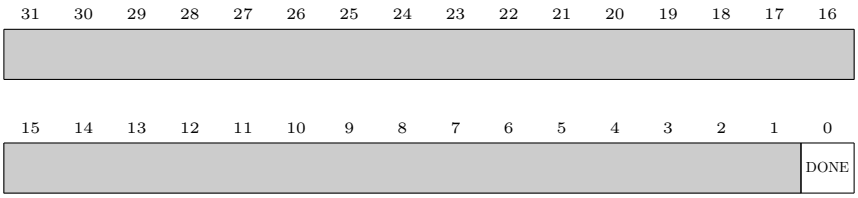
Address: `0x1A10245C`
Reset Value: `0x00000000`



`Bit 0` **START** *(R/W)* Filter start flag, write only, write 1 to start the filter :

4.8.80 REG_STATUS

Address: `0x1A102460`
Reset Value: `0x00000000`



`Bit 0` **DONE** *(R/W)* Filter done flag, write 1 to clear the flag :
-1'b0: Filter process is not finished
-1'b1: Filter process is finished

5 Debug Module for External Debug Support

The debug module in PULPissimo is compliant with the RISC-V External Debug Support specification $v1.13.1$. For more details please take a look at the documentation in the debug module folder and consult the RISC-V External Debug Support specification.