Title: MCU TOP

Abstract: This document is the micro architecture specification and design description of the MCU TOP module.

Prepared by:

Name	Date	Signature
Ignacio Lacadena	20/02/2022	ilacadena

Reviewed by:

Name	Date	Signature
XXX		
XXXX		
XXXXXXX		

Revision History:

REV #	Notes	Date	Authors
0.0	First revision	20/02/2022	Ignacio Lacadena

1 Table of Contents

2 List of tables

3 List of figures

4 MCU_TOP Overview

4.1 High level description

MCU TOP integrates main interfaces and MCU subsystems under one generic hierarchical module.
It is composed by:

- 2 identical MCU subsystems (optionally just one)
- ROM shared by both subsystems
- Peripheral full XBAR that connects MCUs and other host interfaces (a.ka., requestors) with all available resources via control/status register bus (PFL bus).
- In the requestor side, in addition to the two MCU subsytems, the XBAR connects:
    - 1x MDIO/I2C Secondary Interface
    - 1x MDIO Secondary Inteface
    - Each subsystem can access the XBAR using two different paths, allowing simultaneous access from several masters (i.e., MCU or DMA).
- MBOXes to set communications between MCUs or set communications between external host and MCUs
- A semaphore based booking systems so all requestors can soft-lock resources
- Primary SPI, mainly used to download FW from external EEPROM devices
- Global Counter, common to all subsystem.
- AHB bridge so external hosts can have indirect bus access to the MCU's subsystems.
- Configuration and Status registers
- Expansion ports:
    - Two AHB expansion ports (AHB protocol)
    - Parameterizable number of Peripheral expansion ports (PFL protocol)
- Interrupt bus that merges IRQs coming from outside with those generated locally.
- JTAG based debug port with embedded hub to connect several TAP controllers to the same JTAG daisy chain.

4.2 Block-diagram

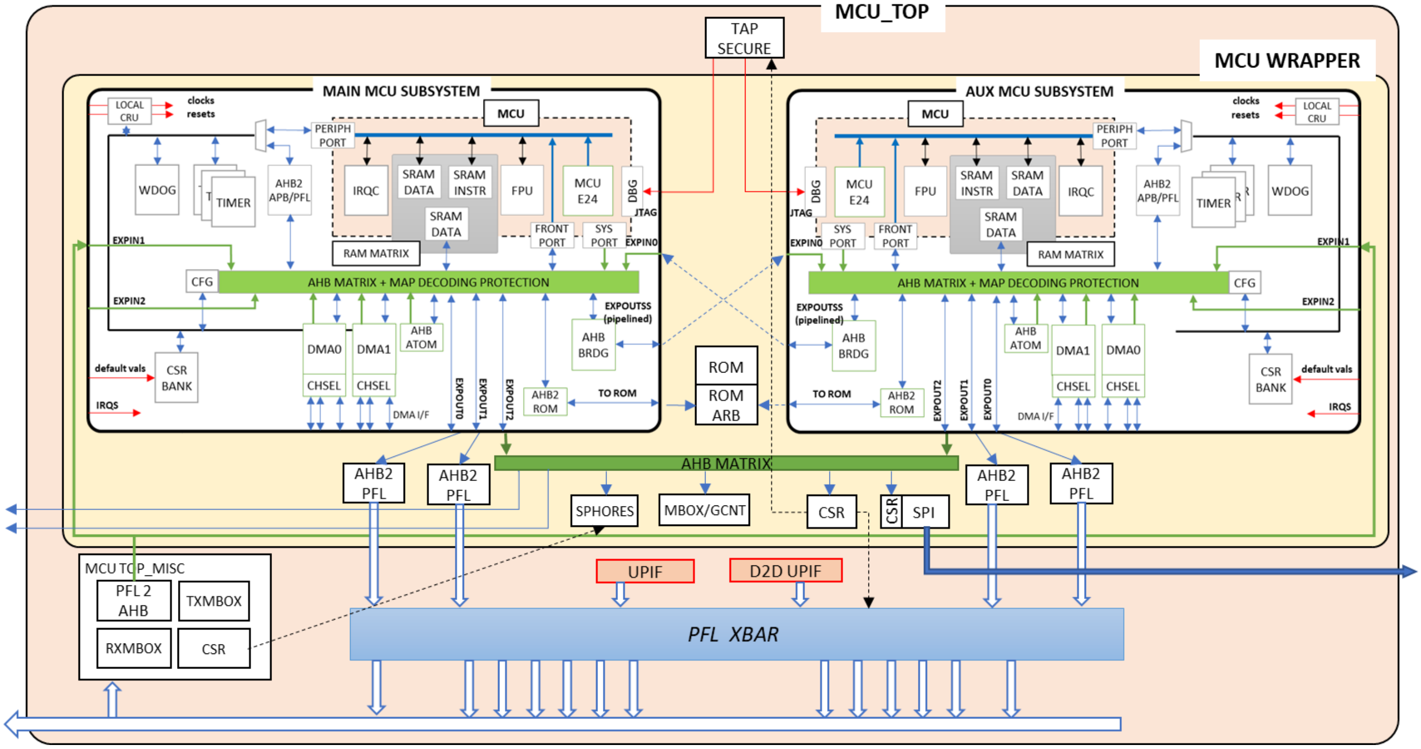

Figure 1: MCU_TOP Block-diagram

5 Portlist and Parameters

5.1 Port List

5.1.1 System ports

Name	Size	Direction	Description
clk	1	input	Main clock.
rtc_toggle	1	input	Real Timer Clock for MCU. Sample as data. Its frequency must be: f(clk) > 2 * f(rtc_toggle)
clk_20m	1	Input	Auxiliar clock used in MDIO and I2C secondary interfaces internal logic
main_mcu_reset_n	1	input	Main MCU reset source (Active L). Must be deasserted only when memprp_done is set to avoid conflicts when accessing memories
aux_mcu_reset_n	1	input	Aux MCU reset source (Active L). Must be deasserted only when memprp_done is set to avoid conflicts when accessing memories
sys_reset_n	1	input	System reset (Active L) Must be deasserted before main/aux_mcu_reset_n. It resets everything except MCU's core reset and the debug logic. When released, it initialize memory erasing process.
clk_soft_reset_n	1	Input	Soft reset going to PFL XBAR related registers and FSM
clk20m_hard_reset_n	1	Input	Async Reset (Active L) synchronized to clk_20m that resets logic in this clock domain inside Secondary interfaces
mdc_hard_reset_n	1	input	Async reset (Active L) of the logic clocked by MDC clock in the MDIO I/F
memprp_done	1	output	Mem initialization done. After releasing sys_reset _n, it will be asserted once memory intitialization has finished (eiter because of not erasing but setting into active state or because of erasing process finished). It is highly recommended not de-asserting reset_n until this flag is asserted.
wdog_rstn_out	1	output	This is the delayed reset output from the wdog (combination of main reset and internal timeout) To be used outside in the main reset generation (optional)
otp_default_bus	DEFAULT_VALUES_W	input	Opt_default values to be used to change default behavior on reset of some registers
otp_default_ok	1	input	When set, opt_default_bus values are valid
self_init_in	1	input	Input pin mapped to internal register tobe used by the FW to distinguish between init modes.

5.1.2 Other ports

Name	Size	Direction	Description
ext_irqs_in	IRQ_EXTIN_W	input	External interrupt bus
ext_irqs_out	IRQ_EXTOUT_W	output	SW Interrupt from MCU going outside
mcu_top_int	1	Output	All in one hierarchical interrupt going out
dma_src_req	DMA_EXT_CH	input	DMA source Request
dma_dst_req	DMA_EXT_CH	input	DMA destination Request
dma_src_done	DMA_EXT_CH	output	DMA source Done
dma_dst_done	DMA_EXT_CH	output	DMA destination Done
gpio_out	GPIO_W	output	General Purpose Lanes coming from registers in MCU subsystem.
gpio_in	GPIO_W	input	General Purpose Lanes going to registers in MCU subsystem where data can beread
cfg_clksel_mcu	1	Output	Configuration ports going out
cfg_clksel_312m	2	Output	Configuration ports going out
cfg_mcu_clk_div	2	Output	Configuration ports going out

5.1.3 Serial I/F ports

Name	Size	Direction	Description
clk_mdc	1	input	UPIF (Host I/F). MDC / I2C SCL
mdio_out	1	output	UPIF (Host I/F). MDIO Output / I2C SDA Output
mdio_out_en_n	1	output	UPIF (Host I/F). MDIO Output Enable / I2C SDA Output Enable (Active L)
mdio_in	1	input	UPIF (Host I/F). MDIO Input / I2C SDA input
upif_i2csel	1	input	UPIF (Host I/F). Mode Selection (0: MDIO, 1: I2C)
upif_phyadr	5	input	UPIF (Host I/F). MDIO Phy Address configration
d2d_scl_in	1	input	Die to Die I/F. I2C Serial Clock
d2d_sda_in	1	input	Die to Die I/F. I2C Serial Input Data
d2d_sda_out	1	output	Die to Die I/F. I2C Serial Output Data
d2d_sda_oe	1	output	Die to Die I/F. I2C Serial Output Enable (Active H)
spi_sclk_out	1	output	Primary SPI. Serial Clock Out
spi_sclk_out_en_n	1	output	Primary SPI. Serial Clock Out Enable (Active L)
spi_sel_out	1	output	Primary SPI. Serial Select Out
spi_sel_out_en_n	1	output	Primary SPI. Serial Select Out Enable (Active L)
spi_miso_in	1	input	Primary SPI. Serial Data In
spi_mosi_in	1	input	Primary SPI. Serial Data In (in Dual mode)
spi_mosi_out	1	output	Primary SPI. Serial Data Out
spi_mosi_out_en_n	1	output	Primary SPI. Serial Data Out Enable (Active L)
spi_wpn_out	1	output	Primary SPI. WP Out
spi_wpn_en_n	1	output	Primary SPI. WP Out Enable (Active L)

5.1.4 Serial I/F configuration ports

Name	Size	Direction	Description
upif_cfg_phyadr_mask	5	input	
upif_cfg_phyadr_value	5	input	
upif_cfg_broadcast_en		input	
upif_cfg_resp_timeout	4	input	
upif_err_cap_rearm		input	
upif_err_cap_address	16	output	
upif_err_cap_mmd_region	2	output	
upif_err_cap_valid		input	
upif_err_cap_wr		input	
upif_cfg_msb_data_x32_from_in		input	
upif_cfg_msb_data_x32_zero		input	
upif_cfg_x16_only		input	
upif_msb_data_reg_in	16	input	
d2d_cfg_resp_timeout	4	input	
d2d_err_cap_rearm		input	
d2d_err_cap_address	16	output	
d2d_err_cap_mmd_region	2	output	
d2d_err_cap_valid		output	
d2d_err_cap_wr		output	
d2d_cfg_msb_data_x32_from_in		input	
d2d_cfg_msb_data_x32_zero		input	
d2d_cfg_x16_only		input	
d2d_msb_data_reg_in	16	input	
pfl_upif_bridge_data_mask	16	output	
we_pfl_upif_bridge_data_mask		input	
wdata_pfl_upif_bridge_data_mask	16	input	
pfl_d2d_bridge_data_mask	16	output	
we_pfl_d2d_bridge_data_mask		input	
wdata_pfl_d2d_bridge_data_mask	16	input	

5.1.5 Expansion I/F ports

Name	Size	Direction	Description
ahb0_haddr	AHB_ADDR	output	AHB Expansion to External Slave 0 I/F
ahb0_hburst	3	output	AHB Expansion to External Slave 0 I/F
ahb0_hclk	1	output	AHB Expansion to External Slave 0 I/F
ahb0_hprot	4	output	AHB Expansion to External Slave 0 I/F
ahb0_hready	1	output	AHB Expansion to External Slave 0 I/F
ahb0_hresetn	1	output	AHB Expansion to External Slave 0 I/F
ahb0_hsel	1	output	AHB Expansion to External Slave 0 I/F
ahb0_hsize	3	output	AHB Expansion to External Slave 0 I/F

ahb0_htrans	2	output	AHB Expansion to External Slave 0 I/F
ahb0_hwdata	32	output	AHB Expansion to External Slave 0 I/F
ahb0_hwrite	1	output	AHB Expansion to External Slave 0 I/F
ahb0_hrdata	32	input	AHB Expansion to External Slave 0 I/F
ahb0_hreadyout	1	input	AHB Expansion to External Slave 0 I/F
ahb0_hresp	2	input	AHB Expansion to External Slave 0 I/F
ahb1_haddr	AHB_ADDR	output	AHB Expansion to External Slave 1 I/F
ahb1_hburst	3	output	AHB Expansion to External Slave 1 I/F
ahb1_hclk	1	output	AHB Expansion to External Slave 1 I/F
ahb1_hprot	4	output	AHB Expansion to External Slave 1 I/F
ahb1_hready	1	output	AHB Expansion to External Slave 1 I/F
ahb1_hresetn	1	output	AHB Expansion to External Slave 1 I/F
ahb1_hsel	1	output	AHB Expansion to External Slave 1 I/F
ahb1_hsize	3	output	AHB Expansion to External Slave 1 I/F
ahb1_htrans	2	output	AHB Expansion to External Slave 1 I/F
ahb1_hwdata	32	output	AHB Expansion to External Slave 1 I/F
ahb1_hwrite	1	output	AHB Expansion to External Slave 1 I/F
ahb1_hrdata	32	input	AHB Expansion to External Slave 1 I/F
ahb1_hreadyout	1	input	AHB Expansion to External Slave 1 I/F
ahb1_hresp	2	input	AHB Expansion to External Slave 2 I/F
pfl_mack	MCUTOP_EXT_PFL	input	PFL expansion I/F: Acknowledge
pfl_mdataout	MCUTOP_EXT_PFL* DATA_W	input	PFL expansion I/F: Read Data bus
pfl_merr	MCUTOP_EXT_PFL	input	PFL expansion I/F: Error response
pfl_clken	MCUTOP_EXT_PFL	output	PFL expansion I/F: Clock Enable
pfl_maddr	MCUTOP_EXT_PFL* ADDR_W	output	PFL expansion I/F: Peripheral Address bus
pfl_mdatain	MCUTOP_EXT_PFL* DATA_W	output	PFL expansion I/F: Write Data bus
pfl_msel	MCUTOP_EXT_PFL	output	PFL expansion I/F: Peripheral Select (Active H)
pfl_mwr	MCUTOP_EXT_PFL	output	PFL expansion I/F: Peripheral write
cfg_bcast_en	MCUTOP_EXT_PFL	input	PFL expansion: Broadcast Mask Configuration
i_cfg_clken_cnt	2	input	PFL expansion: Clock Enable Configuration
i_sel_pfl_latency	log2(MARGIN_READ_LATENCY)	input	PFL expansion: Latency config
i_pfl_local_latency_adj	log2(MARGIN_READ_LATENCY)	input	PFL expansion: Latency config

5.1.6 DFT Ports

Name	Size	Direction	Description
test_mode_en	1	input	Test mode to put local resets and clock gaters into bypass (when set)
i_scan_rstn	1	input	Reset signal to be used when in test mode (active low)
dft_bypass_mock	1	input	Input port to be used during TAP (dft) insertion. Must be tied to 0. (please force to ‘1’ to assist DFT tools to infer JTAG hookup)
jtag_tck	1	input	JTAG clock. Typically coming from primary IOs
jtag_tms	1	input	JTAG TMS. Typically coming from primary IOs

jtag_tdi	1	input	JTAG Serial In. Typically coming from primary IOs
jtag_trstn	1	input	JTAG TRSTn (Active L). Typically coming from primary IOs
jtag_tdo	1	output	JTAG Serial Out. Typically going to primary IOs
jtag_tdo_oe_n	1	output	JTAG Serial Output Enable (Active L). Typically going to primary IO
jtag_mfr_id	11	input	Manufacturer ID field of the Chip-ID
jtag_part_number	16	input	Part Number field of the Chip-ID
jtag_version	4	input	Version fieldof the Chip-ID
dft_tck	1	output	JTAG TCK. To be connected to Test Tap Controller
dft_tms	1	output	JTAG TMS. To be connected to Test Tap Controller
dft_tdi	1	output	JTAG TDI. To be connected to Test Tap Controller
dft_trstn	1	output	JTAG TRSTn. To be connected to Test Tap Controller
dft_tdo	1	input	JTAG TDO. To be connected to Test Tap Controller
dft_tdo_oe_n	1	nput	JTAG TDO output enable (Active L). To be connected to Test Tap Controller

5.2 Parameters

Name	Type	Default	Description
IRQ_EXTOUT_W	int	2	Number of SW interrupts controlledby MCU going outside.
IRQ_EXTIN_W	int	81	Number of external interrupts coming in
DMA_EXT_CH	int	12	Number of DMA channels
DEFAULT_VALUES_W	int	128	Width of the default values bus (caoncatenated all together) to change reset behavior of some specific registers
GPIO_W	int	4	Number of GPIO lanes going out/coming in to MCU_TOP registers
AHB_ADDR	int	16	Address bus width going out through expansion I/F
MCUTOP_EXT_PFL	int	40	Number of PFL resources to be connected from outside
DATA_W	int	32	PFL Data width
ADDR_W	int	16	PFL Address width
MARGIN_READ_LATENCY	Int	8	PFL Latency

Table 1: Parameters

5.3 defines

This section lists all defines included in the main subsystem that can be modified (or defined) in order to change RTL behavior for debugging or simulation purposes.

Name	Description
SP5N_SIM_DONT_ERASE_RAM	If defined, it will change the default behavior of the block erase at reset by not erasing any memory content. This is intended to save simulation time. Must NOT be used during actual simulations or synthesis
MEM_BEHAVIORAL	If set, actual memory macros will be replaced by a behavioral model that can infer memories when synthesizing into FPGA
ROM0=<path_to_file>	Add this define to add content to ROM0
ROM1=<path_to_file>	Add this define to add content to ROM1
SYNTHESIS	This define is usually utilized by Synthesis tools to emulate translate on/off
MCU_ROM_USE_BIST_CFG	If set, it will add corresponding DFT ports

Table 2: Defines

6 Memory Map

There are three different memory maps depending on the access point. These are:

- MCU: MCUs has visibility of the full map, including those memories and block that are internal to the MCU Core. Base addresses and, in particular, all configuration register location are all word aligned. And special case is DMA or ATOM accesses. These AHB masters have restricted access to blocks inside the MCU core, allowing just the access to the internal memories.
- PUF / D2D-PUF: These two access point goes directly to the PFL Xbar. PFL XBAR accesses are all of te same size and address bus is aligned to LSB.
  These access points have'nt direct visibility of MCU subsystems' memory map. In orderto access it, it requires a indirect access using the PFL2AHB bridge located in the MCU_TOP register map space.
- PFL2AHB: This list access point is connected to one of the master inputs of the AHB Matrix. In the same way than the DMA or ATOM, it cannot access addresses inside MCU core except for the RAM and DRAM.

6.1 MCU Internal Memory Map

This map is only visible from MCU Cores and JTAG access.

Base Addr	Top Address	PMA*		Slave Name
0x0000_0000	0x0000_0FFF			Debug
0x0000_1000	0x0000_2FFF			Reserved
0x0000_3000	0x0000_3FFF	RWX	A	Error Device
0x0000_4000	0x016F_FFFF			Reserved
0x0170_0000	0x0170_0FFF	RW		Bus Error Unit
0x0170_1000	0x01FF_FFFF			Reserved
0x0200_0000	0x02FF_FFFF	RW	A	CLIC
0x0300_0000	0x1FFF_FFFF			Reserved
0x2000_0000	0x2FFF_FFFF	RWX	A	Periph-Port / MCU-APB
0x3000_0000	0x4FFF_FFFF			Reserved
0x5000_0000	0x5007_FFFF	RWX	A	TIM0
0x5008_0000	0x500F_FFFF	RWX	A	TIM1
0x5010_0000	0x7FFF_FFFF			Reserved
0x8000_0000	0xFFFF_FFFF	RWX		Sys Port / AHB Subsystem
* Physical Memory Attributes: R–Read, W–Write, X–Execute, A–Atomics				

Table 3: MCU Internal Memory Map

6.2 Full AHB Memory Map

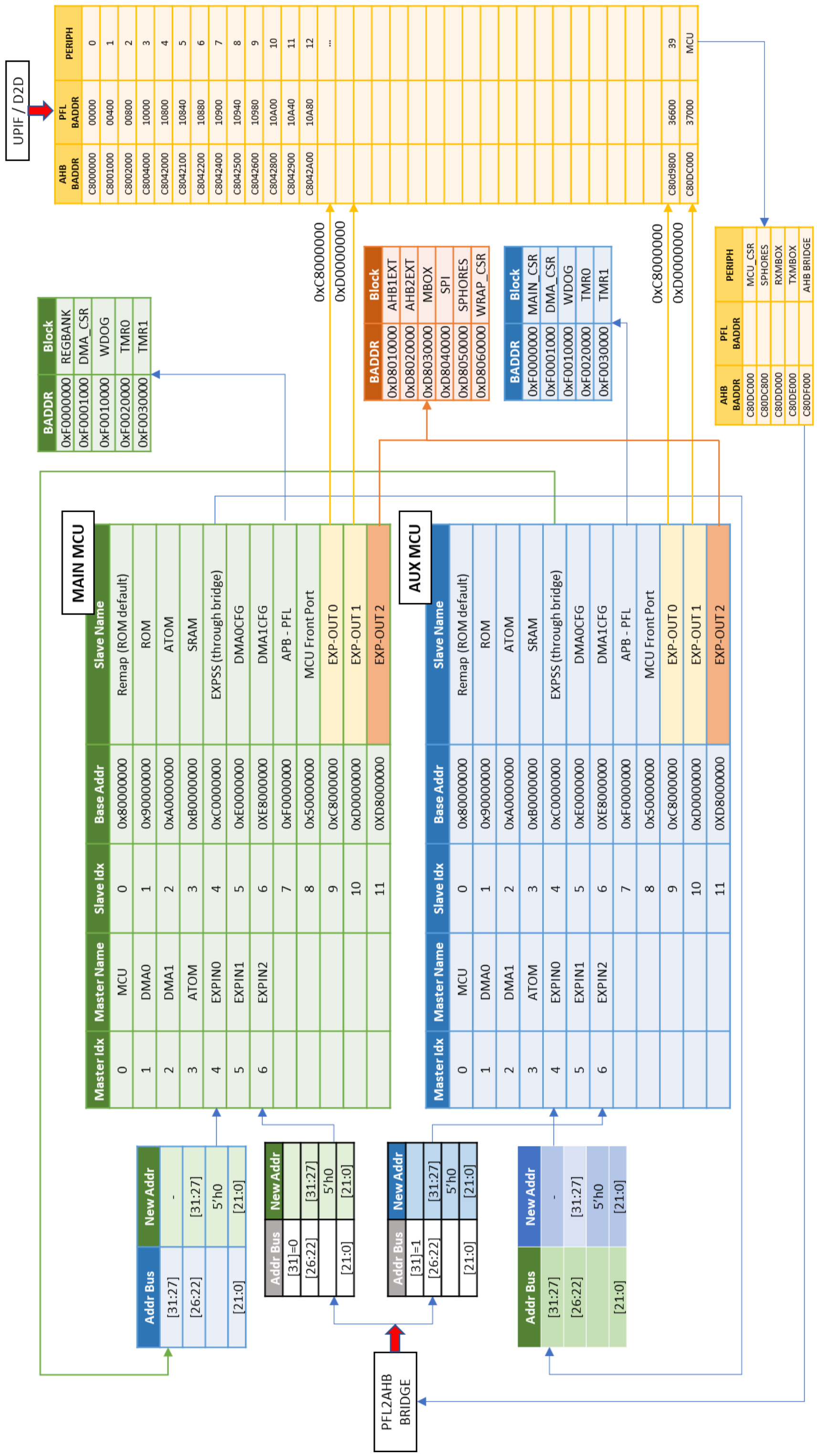

Table 4: MCU Memory Map

6.2.1 AHB Memory Map

Once in the AHB subsystem, regardless from which master the AHB is accessed to, the memory map that is seen is described in the following table. Remap Base Address is either 0x00000000 and 0x80000000

Master Idx	Master Name	Slave Idx	Base Addr	Slave Name
0	MCU	0	0x00000000 0x80000000	Remap (ROM default)
1	DMA0	1	0x90000000	ROM
2	DMA1	2	0xA0000000	ATOM
3	ATOM	3	0xB0000000	SRAM
4	EXPIN0	4	0xC0000000	EXPSS (through pipelined bridge)
5	EXPIN1	5	0xE0000000	DMA0CFG
6	EXPIN2	6	0XE8000000	DMA1CFG
		7	0xF0000000	APB - PFL
		8	0x50000000	MCU Front Port
		9	0xC8000000	EXP-OUT 0
		10	0xD0000000	EXP-OUT 1
		11	0XD8000000	EXP-OUT 2

Table 5: AHB Memory Map

Please note that due to bridges across systems, there are some addresses conversion that make a given master not to have full access to all slaves.

6.2.2 Addresses Conversions

As can be depicted from the Full AHB map figure, there are some conversions that must be observed.

- Addresses conversion through EXPSS
  This conversion happens when a master in one subsystem accesses the other subsystem. This is allowed for the following masters: MCU, DMA0, DMA1, ATOM. This is forbidden for PFL2AHB and “the other subsystem”.
  The conversion follows the next rules:
    - ADDR[31:27] are discarded
    - ADDR[26:22] are moved to the NEW_ADDR[31:27]
    - ADDR[21:0] are the same in NEW_ADDR[21:0]
    - NEW_ADDR[26:22] is always 0

- Addresses conversion through PFL2AHB
  This conversion happens when using indirect access from PFL2AHB bridge.
  In this case, the conversion follows the next rules:
    - ADDR[31] decides if accessing MAIN_MAP or AUX_MAP
    - ADDR[26:22] are moved to the NEW_ADDR[31:27]

- ADDR[21:0] are the same in NEW_ADDR[21:0]
- NEW_ADDR[26:22] is always 0

6.2.3 Full map and reg description

Please refer to the register database for detailed information.

- mcu_reg.db: memory map as it is seeng from MCUs
- fe_reg.db: memory map as it is seeng from PFL-XBAR
- pfl2ahb_reg.db: memory map as it is seeng from PFL2AHB bridge

7 Interrupts

SiFive MCU has two type of interrputs:

- 127 Local interrupt Bus, for which 16 bits are internal to Sifive.
- 1 MEIP interrupt

Please refer to SiFive documentation for further information

In mcu_top most of the HW and SW sources are concatenated to provide the local interrupt bus.

MEIP is also used, handlng by SW by writing to the corresponding registers,

7.1 MCU Local Interrupts

The MCU has been configured to support up to 127 interrupts. The first 16 are reserved for MCU internal use.

Some other are internal to each subsystem, some of them are generated at mcu_top level.

The complete map seen at mcu_top level is the following.

Source	Offset	Number of IRQs	Comment
Internal to Sifive	0	16	Interrupt that are internal to MCU. Please, check reference for further information
dma0	16	4	One IRQ per channel
dma1	20	4	One IRQ per channel
wdog	24	1	
ahb_subsystem	25	1	For memory errors from several sources
rsv	26	1	
Timer 0	27	1	Interrupt from timer 0
Timer 1	28	1	Interrupt from timer 1
rsv	29	3	Not used but can be set by FW
AHB MBOX	32	1	Interrupt coming from the MBOX. This interrupt is different for each subsystem.
GLB TMR	33	1	Global Timer interrupt. This interrupt is different for each subsystem.
SPI	34	1	Primary SPI interrupt
Rsv	35	1	

Pfl_aerr	36	1	PFL Address Error Interrupt from several sources
RX/TX MBOX	37	1	RX / TX Mbox interrupt
Pfl_write_err	38	1	PFL XBAR Write error Interrput
Host2mcu0	39	1	Interrupt coming from MCU_TOP.HOST2MCU0 register
Host2mcu1	40	1	Interrupt coming from MCU_TOP.HOST2MCU1 register
External	41	81	Interrupt Bus coming from EXT_IRQ_IN port.

Table 6: IRQ Bus

All sources are concatenated together and resynchronized to the system clock. The local bus can be intercepted by FW by overwritten their values in the corresponding registers of the CSR register bank.

External interrupts are considered to be Active High levels.

7.2 MCU Machine External Interrupt

There are two registers in the MCUJ_TOP mempry area in charge of asserting the MEIP input of each subsystems: one register per subsystem.

The registers are located in the MCU_TOP mem map and are accessible via PFL-XBAR.

7.3 Interrupt to Host

MCU_TOP has the following output ports:

- EXT_IRQS_OUT: These interrupts are coming from SW register inside the main subsystem. There is one interrupt for each subsystem
- MCU_TOP_INT: This interrupt merges several sources into just one interrupt going outside (typically to the external host). The sources of this interrupt are the following:
    - HCPU_MAIN. This is another hierarchical interrupt, It groups several status signals (total of 6) coming from the main mcu:
        - MCU's cease, halt, wfi, debug
        - Main_memprp_done
        - EXT_IRQ
    - HCPU_AUX. This is another hierarchical interrupt, It groups several status signals (total of 6) coming from the main mcu:
        - MCU's cease, halt, wfi, debug
        - Main_memprp_done
        - EXT_IRQ
    - MBOX-INT: Coming from TX/RX-MBOX
    - MCU2HOST0 and MCU2HOST1: These interrupts are asserted when the value of corresponding register is different from 0

8 Default bus

The are some registers which reset value can be overwritten by information available before reset (like values coming from fuses).

Just after releasing the reset signal, the otp_default_bus and otp_default_ok ports are sampled. If otp_default_ok is asserted, then the default reset values of the registers will be replaced by those in otp_default_ok.

In addition, a 0 → 1 toggle in the otp_default_ok will also for the registers to re-load their content with the information convey in the otp_default_bus

These registers are:

Configuration Name	OTP-Bus bits Main / Aux		Register Default	Description
mcu_rst_vector	[31:0]	[95:64]	0x80000000	Address at which MCU start execution after reset
mcu_rst_vector_ok	[32]	[96]	-	Need to be set to replace mcu_rst_vector
dram_num	[37:33]	[101:97]	4	Number of block elements assigned to dram / tim1
sram_num	[42:38]	[106:102]	1	Number of block elements assigned to sram / ahb
iram_num	[47:43]	[111:107]	8	Number of block elements assigned to iram / tim0
ram_num_ok	[48]	[112]	-	Need to be set to replace dram/iram/sram_num
init_erase	[61:49]	[125:113]	13'h1EFF	If set, the corresponding block will be erased at reset.
Init_erase_ok	[62]	[126]	-	Need to be set to replace init_erase field

Table 7: OTP Default Bus

Please note OTP default settings could be different for each subsystem. Hence

- OTP_BUS[63:0] will be connected to Main_subsystem
- OTP_BUS[127:64] is connected to Aux Subsystem

9 Design Specifications

9.1 Integration matters

9.1.1 Integration details

The following description would help to understand how the subsystems and other blocks have been integrated into MCU_TOP.

Let's remind how mcu subsystems had been integrated into mcu_wrapper, first:

1. Expansion buses of both subsystems have been connected in the same way:
   a. Expout0 and expout1 are connectedto AHB2PFL-Requestor bridges.
   b. Expout2 behaves as a master in the small AHB matrix to access shared AHB resources.
   c. ExpoutSS were connected to Expin0 of the other subsystem.
   d. ROM I/F are both accessing the same ROM block. An arbiter is in charge of guaranteeing even access from each subsystem.
   e. Expin1 and expin2 comes from upper levels.
2. Shared AHB resources are:
   a. 2x expansion AHB I/F going to upper levels
   b. Semaphore block in charge of registering resources utilization. Up to 64 sempahores are available. It also has a direct port (not AHB) to enable external blocks to reserve / query resources uytilization.
   c. 8x MBOX registers and Global counters that are able to generate independent interrupts to each subsystem
   d. Primary SPI with EEPROM protocol support
   e. CSR register bank. Some of the configurations are used outside in upper levels
3. Debug I/F, OTPs, some system or status signals were brought to ports. The decision of whether these buses are shorted or not together was pushed to upper levels.

At MCU_TOP, the following decisions were made:

1. Expin1 of each subsystem is left unsued. Expin2 is the gateway for indirect accesses coming from PFL-XBAR (UPIF and D2D-PIF).
2. OTP Bus is not shared. Each subsystem requires 64-bits
3. In addition to mcu_wrapper, the following blocks had been integrated:
   a. MDIO/I2C Secondary Interface (intended to be connected to an external host)
   b. MDIO Secondary interface (intended to connect another MCU_TOP for a die-to-die communication in a SiP.
   c. All these and all the AHB2PFL-REQ bridges are connected to the main PFL-XBAR so the 6 requestors can accesses a parameterized number of PFL resources.
   d. Most of the PFL resources will be instanatiated in upper levels. However, MCU_TOP has the following service resources already:
      i. TX/RX-MBOX (FIFO based) to set communications between MCU's and external Hosts/devices.
      ii. PFL2AHB bridge so accesses through the PFL-XBAR (mainly from UPIF and D2D-PIF) can have access to lamost all the memory space under AHB subsystem.
      iii. Configuration and status registers,
      iv. Access to sempahores located inside mcu_wrapper

9.1.2 Main and Aux Subsystems Differences

MCU TOP integrates two identical subsystems in a symmetric manner: any action from one subsystem can be replicated by the other. However, there are some specific settings / hookup that are custom for each subsystem that has to be considered:

- System Signals:

  Clocks and system reset are common to both subsystems. However, there is an independent reset going to each mcu_core, respectively.

  In the opposite direction, each subsystem has an independent memprp_done signals. Both signals are combined (logic and) to generate the memprp_done signal that is sent outside

- PFL-XBAR connection:

  Main subsystem is connected to PFL-XBAR requestors 0 and 1.

  Aux subsystem is connected to PFL-XBAR requestors 4 and 5.

- MEIP connection:

  Main subsystem has its MEIP signal coming from MCU_MAIN_MEIP register

  Aux subsystem has its MEIP signal coming from MCU_AUX_MEIP register

- Subsystem ID:

  There is a register in Subsystem regiser bank, called AHB_APB_PFL_REGBANK_system_info, that its value is different. This way, FW can identify in which subsystem is running and take the correct branch:
    - MAIN Subsystem ID: 0xDEADFACE ;
    - AUX Subsystem ID: 0xBEADCAFE ;

- JTAG Chip ID:

  Although both MCU's debug ports share the same ID with the Test TAP controller, the version numer has been changed in order to identify each TAP controller.

  Main MCU Chip ID version is obtained with jtag_version[3:0] ^ 4'h8

  Aux MCU Chip ID version is obtained with jtag_version[3:0] ^ 4'h4

  In the daisy chain, Main MCU is placed first (closer to TDI); Aux MCU comes later (the last TAP controller in the chain).

9.1.3 Forbidden paths

Due to integration decisions, the following paths had been blocked:

1. PFL2AHB cannot access PFL-XBAR, avoiding loops.
   The intention is to preven an access generated from UPIF, that goes through PFL-XBAR going to PFL2AHB; from there, an indirect access can reach AHB matrix in the subsystem accessing through expin2 and going to either Expou0 or Expout1.
2. PFL2AHB cannot access a subsystem using ExpoutSS.
   PFL2AHB should decide which subsystem want to access by setting ADDR[31].

3. Expin0 of a subsystem (that is connected to the other subsystem) cannot access ExpoutSS, that would connect to the original AHB matrix.

The following paths, although possible, should neve been used.

4. Starting from one subsystem, going to the other subsystem (using expoutSS → expin0) and accessing PFL-XBAR. PFL-XBAR should only be accessed from expin0 and expin1 of the same subsystem.

9.2 PFL XBar

The PFL XBAR is a 6 to 41 interconnection matrix.Arbitration is done at the resource size, allowing accesses in parallel without waits when requestors are not entering into conflict.

The PFL-XBAR support 32-bits of data, for both write and read. In the write direction the following type of accesses are assumed, depending on the resource:

- Full 32-bits
- plain 16-bits
- bit-masked 16 bits (16-bits for data, 16-bits for mask)

On read, if the useful data bits are 16-bts only, they upper half-word will read all zeros.

Refer to register database to learn about base addresses and their decoding

9.2.1 Hookup and Requestor's priority

PFL-XBAR has 6 requestors. They have been connected in the following way:

1. Main subsystem, expout0
2. Main subsystem, expout1
3. UPIF
4. D2D-PIF
5. Aux subsystem, expout0
6. Aux subsystem, expout1

When the same resource is accessed by several requestors, the following priority table sall be applied:

1. UPIF (highest priority)
2. Main subsystem, expout0
3. Aux subsystem, expout0
4. D2D-PIF
5. Main subsystem, expout1
6. Aux subsystem, expout1 (lowest priority)

9.2.2 Protection

By default, all requestors can freely access every resources.

In order to restrict which resources can be used by whom, each requestor I/F has assigned a 11-bits configuration register to disable paths going through the PFL XBAR. These 11 bits are:

Disable_bus offset	Resource Name Locked
0	MMD08
1	MMD30 CORE
2	MMD30 DP TOP
3	MMD30 RSDEC
4	MMD30 ERU_OIMC
5	MMD30 MCU
6	MRX
7	ORX
8	MTX
9	OTX
10	CLKMON TX

9.3 Access to Semphores

The Semaphores block allows several hosts or requestor to be aware if another requestor is using a given resource.

When any host wants to use a shared resource, if should first check if its assigned semaphore is locked or not. If not, the semaphores will be locked and the host can take control of the resource. After it finishes, it should release th lock.

There isn't any actual HW locking/unlocking resources accesses. Semaphore blocks is for information only. The semaphore policy requires a FW protocol on top so host can fairly wait until it's lock is released. A tag system has been added to identify who is locking a given resource.

Semaphore block has two interface: AHB and direct (from PFL)

9.3.1 From Subsystems (AHB I/F)

This is the fastest way to access (no latency). Since the AHB I/F is connected to the small matrix in mcu_wrapper, arbitration is done at the AHB-matrix level

9.3.2 From MCU_TOP CSR

The direct / PFL interface has been connected to some registers in the MCU_TOP register area.IT basically works in the same way as the AHB I/F except for arbitration.

There isn't any arbitration when maniopulating sempahores fro the MCU_TOP CSR. Instead, a flag is asserted when a conflict happens because of simulatenous access. Host should check the flag and retry in the case the access failed (was busy).

9.4 TAP Security

Tap secure block is in charge of

a) restricting the access to transaction
b) interconnects all TAP controllers so as the JTAG host

9.4.1 Changing default behavior

The default behavior is obtained from parameters of the block. In order to change the default behavior

A new configuration (including cfg_ok=1) must be overwrritem before activity is detected in TMS signal

9.5 SPI

A Primary SPI is connected to the AHB small matrix (mcu_Wrapper) and is shared among subsystems and PFL2AHB bridge.

9.5.1 Memory maps

There are three main areas in the SPI memory map:

- Data buffer 0. Data is stored in buffer 0 when buffer one is full
- Data buffer 1. Data is stored in buffer 0 when buffer one is full
- Regiser bank, where configuration and commands can be set

9.5.2 IO configurability

All SPI IOs can be configured to be push-pull or open-drain.

9.5.3 DMA support

TBD

9.6 Clock and Resets

This section describes how clocks and resets are handled inside main subsystem.

9.6.1 Main system signals

The primary system signals are the following

- System clocks and resets (clk)
  - Clock (clk)
    All blocks inside the subsystem are synchronous to this clock (or a gated version of it). Its maximum frequency is 300MHz (TBD)
  - rtc_toggle
    This is not a real clock but a periodic signal which period must be at least two times the period of the system clock. It is needed by the MCU and is considered (sampled) as data.
  - System reset (reset_memprp_n)
    The reset is active low and and it is internally synchronized to system clock. It is in charge of generating all reset signals except the one going to the MCU. Apart of this, there are two actions that are automatically launched when this reset is released:
    - Memory initialization is started ater releasing this reset and it can be either setting blocks into active state or completely erase their contents. Whether to erase or not a block ram is decided by the init_erase register in the regbank.

- OTP default bus sampling:
  Just after releasing the reset, otp_default_bus and otp_default_ok ports are sampled.

- mcu reset (reset_n)
  The reset is active low and it is internally synchronized to system clock. This reset is in charge of generating the correct reset waveforms going to the MCU. In order to avoid conflicts with memory initialization, it is recommended not de-asserting it until memory initialization has finished (signalled with the assertion of memprp_done)
- memprp_done (output)
  This signal is asserted once all block memories have been initialized.

9.6.2 Local management

There isn't any local management of clocks and resets at MCU_TOP.

- Clock gating can be done in upper levels or at subsystem level
- Reset gating can be done in upper levels or at subsystem level.

Only memprp_done signal can be managed locally. Memprp_done outputs of each subsystem can be used as is or intercepted and overriding by SW.

9.7 DMA Channel selection

TBD

10 Links to other specs

Block	Reference link
Sifive	Manual.pdf User_guide.pdf
AHB Matrix	70107_IPC-AhbFabric-AHB_Datasheet.pdf
DMA	70119_IPC-DMA4-AHB_Datasheet.pdf
WDOG	70125_IPC-Wdog-APB_Datasheet.pdf
TIMER	70124_IPC-Timer-APB_Datasheet.pdf
AHB_ATOM	MAS_AHB_atom_v1p0.docx
AHB2APB	MAS_AHB2APB_v1.2.doc
RAM Wrapper	MAS_RAM_Wrapper_v1.1.doc
SRAM Matrix	MAS_SRAM_Matrix_v1.1.doc
REGBANK	mcu_reg.db / fe_reg.db / pfl2ahb_reg.db
Main Subsystem	Main_ss_v0p0.pdf