Register file structure : regfile_pcie2AxiMaster.pdf
Created by amarchan on 2020/09/14 14:50:02
Register file CRC32 : 0x9CA38D67

1. Main Parameters

Register file endianness: little endian
Address bus width: 10 bits
Data bus width: 32 bits

2. Memory Map

Section name	Address(es) / Address Ranges	Register name	Access Type
info	0x000	tag	R
	0x004	fid	R
	0x008	version	R
	0x00C	capability	R
	0x010	scratchpad	RW
fpga	0x020	version	R
	0x024	build_id	R
	0x028	device	R
	0x02C	board_info	R
interrupts	0x040	ctrl	RW
	0x044, 0x048	status (1:0)	RW
	0x04C, 0x050	enable (1:0)	RW
	0x054, 0x058	mask (1:0)	RW
interrupt_queue	0x060	control	RW
	0x064	cons_idx	RW
	0x068	addr_low	RW
	0x06C	addr_high	RW
tlp	0x070	timeout	RW
	0x074	transaction_abort_cntr	RW
spi	0x0E0	SPIREGIN	RW
	0x0E8	SPIREGOUT	R
arbiter	0x0F0	ARBITER_CAPABILITIES	R
	0x0F4, 0x0F8	AGENT (1:0)	RW
axi_window [0]	0x100	ctrl	RW
	0x104	pci_bar0_start	RW
	0x108	pci_bar0_stop	RW
	0x10C	axi_translation	RW
axi_window [1]	[0x110 - 0x11C]	...	...
axi_window [2]	[0x120 - 0x12C]	...	...
axi_window [3]	[0x130 - 0x13C]	...	...
debug	0x200	input	R

Section name	Address(es) / Address Ranges	Register name	Access Type
	0x204	output	RW
	0x208	DMA_DEBUG1	RW
	0x20C	DMA_DEBUG2	RW
	0x210	DMA_DEBUG3	R

Section: info — PCIe2AxiMaster IP-Core general information

Address Range: [0x000 - 0x010]

Description:
This section contains all the register related to the IP-Core identification and capability listing.

tag — Matrox Tag Identifier

Address: section "info" base address + 0x000

Description:
This register contains the Matrox tag identifier string. Very convenient in debug mode for identifying the IP-Core register space.

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

value (23:0)	Tag value	
STATIC	This is a 3 character string. The value is "MTX"	
Value at Reset:	0x58544d	
Possible Values:	0x58544D	MTX ASCII string

fid — Matrox IP-Core Function ID

Address: section "info" base address + 0x004

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

value (31:0)	
STATIC	
Value at Reset:	0x0

version — Register file version

Address: section "info" base address + 0x008

Description:
Register file version composed of 3 sub-fields
Major version
Minor version
sub-minor version

v0.1.0 : First registerfile revision
v0.2.0 : Added the fpga/board_info register

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

major (7:0)	Major version	
STATIC	Indicates a major register file change that breaks software compatibility.	
Value at Reset:	0x0	
Possible Values:	Any Value	

minor (7:0)	Minor version	
STATIC	Indicates a minor register file change that do not break software compatibility.	
Value at Reset:	0x9	
Possible Values:	Any Value	

sub_minor (7:0)	Sub minor version	
STATIC	Indicates	
Value at Reset:	0x0	
Possible Values:	Any Value	

Address: section "info" base address + 0x00C

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

value (7:0)	
STATIC	
Value at Reset:	0x0

scratchpad — Scratch pad

Address: section "info" base address + 0x010

Description:

R/W software debug register. Writing or reading to that register has no effect on the hardware.

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

value (31:0)		
RW		
Value at Reset:	0x0	
Possible Values:	Any Value	

Section: fpga — FPGA informations

Address Range: [0x020 - 0x02C]

version — Register file version

Address: section "fpga" base address + 0x000

Description:
Register file version composed of 3 sub-fields
Major version
Minor version
sub-minor version

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

firmware_type (7:0)	Firmware type	
RO		
Possible Values:	0x0	Driver update
	0x1	NPI Golden firmware
	0x2	Engineering firmware
	Others	Reserved

major (7:0)	Major version	
RO	Indicates a major register file change that breaks software compatibility.	
Possible Values:	Any Value	

minor (7:0)	Minor version	
RO	Indicates a minor register file change that do not break software compatibility.	
Possible Values:	Any Value	

sub_minor (7:0)	Sub minor version	
RO	Indicates	
Possible Values:	0x1	

build_id — Firmware build id

Address: section "fpga" base address + 0x004

Description:
The build ID is a unique incrementing 32 bits number used to identify an FPGA firmware. This value is simply the Unix time stamp (Unix Epoch)

Unix time (also known as POSIX time or UNIX Epoch time) is a system for describing a point in time, defined as an approximation of the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.

For more info https://en.wikipedia.org/wiki/Unix_time

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

value (31:0) *RO*		
Possible Values:	Any Value	

device

Address: section "fpga" base address + 0x008

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

id (7:0)	Manufacturer FPGA device ID	
RO	Lookup table providing the FPGA device ID. The value is user defined (project specific) and specified outside of this document.	
Possible Values:	Any Value	Define outside of this document

board_info — Board information

Address: section "fpga" base address + 0x00C

Description:
This register report board specific information

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

capability (3:0)	Board capability	
RO	Report the board capability (Connected to the board strapping)	
Possible Values:	0x0	2 ToE Ports available
	0x1	4 ToE ports available
	Others	Reserved

Section: interrupts

Address Range: [0x040 - 0x058]

Description:

Mapping provided in the system instantiating this IP.

ctrl

Address: section "interrupts" base address + 0x000

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

num_irq (6:0)	Number of IRQ
RO	Indicated the total number of IRQ connected to the pcie2AxiMaster

global_mask	Global Mask interrupt	
RW		
Value at Reset:	0x1	
Possible Values:	0x0	Any enabled interrupt will bi signaled to the host
	0x1	No active interrrupt is signaled to the host

status (1:0) Interrupt status register

Address: section "interrupts" base address + 0x004 + (index * 0x4)

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

value (31:0)	
RW2C	
Value at Reset:	0x0

enable (1:0) Interrupt status enable

Address: section "interrupts" base address + 0x00C + (index * 0x4)

Description:
Enable the the interrupt status register. If an enable bit is set to 1, the detection of an event is latch in the associated status reister bit.

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

value (31:0)	
RW	
Value at Reset:	0x0

mask (1:0) Interrupt event mask

Address: section "interrupts" base address + 0x014 + (index * 0x4)

Description:
Mask the interrupt event at the source. A masked interrupt is not latched in the status register in legacy interrupt mode, neither is it forwarded to the host by the interrupt queue mechanism.

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

value (31:0)	
RW	
Value at Reset:	0xffffffff

Section: interrupt_queue

Address Range: [0x060 - 0x06C]

Description:
This section controls the behavior of the interrupt queue

control

Address: section "interrupt_queue" base address + 0x000

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

nb_dw (7:0)	Number of DWORDS
STATIC	This is the number of 32-bit DW used to represent all interrupt sources. It is used by the driver to know how to split the data of the interrupt queue in interrupt events. This number should always be a power of 2 to simply the hardware implementation and avoid having a single interrupt event split by the wrap-around boundary.
Value at Reset:	0x1

enable	QInterrupt queue enable
RW	This bit is used to enable the interrupt queue. When disabled, the interrupt will behave in a legacy way where all interrupts are merged into interrupt status register and driver has to read the status register to know the interrupt sources. To reset the interrupt queue, the driver should disable the queue and re-enable it. This will cause the producer index to be reset to 0 internally in the hardware. The driver should write the whole queue area to 0 to make sure it does not mis-interpret the data in the queue as events when the queue is turned back to on.
Value at Reset:	0x0

cons_idx Consumer Index

Address: section "interrupt_queue" base address + 0x004

Description:
The consumer index indicates up to which element of interrupt queue array it can write. Element in the queue between CONS_IDX (included) and PROD_IDX (not included) belong to the driver and are not written by the hardware.

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

cons_idx (9:0)	
RW	When turning on the interrupt queue, the driver should first write this index to value 1023 (0X3FF) to indicate that the queue is empty.
Value at Reset:	0x0

addr_low

Address: section "interrupt_queue" base address + 0x008

Description:
This is the lower part of the address in host memory where the PCIe device writes the interrupt queue. It has to be aligned on 4K bytes boundary.

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

addr (31:0) *RW (31:12)* *RO (11:0)*	
Value at Reset:	0x0

addr_high

Address: section "interrupt_queue" base address + 0x00C

Description:
This is the high part of the address in host memory where the PCIe device writes the interrupt queue. It must be written to 0 if the queue resides in the first 4 GB of memory.

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

addr (31:0)	
RW	
Value at Reset:	0x0

Section: tlp — Transaction Layer protocol

Address Range: [0x070 - 0x074]

Description:
This section contains the registers related to the TLP transactions logic.

timeout — TLP transaction timeout value

Address: section "tlp" base address + 0x000

Description:
Set the time out value.

When a transaction is initiate the counter is incremented at every. Each count tick is 16 ns. The reset value is 500 ms

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

value (31:0)	TLP timeout value	
RW	Units are in clock tick. 1 Clock tick = 16 ns.	
Value at Reset:	0x1DCD650	
Possible Values:	0x1DCD650	500 ms

transaction_abort_cntr TLP transaction abort counter

Address: section "tlp" base address + 0x004

Description:
This register calculate the number of transaction that aborted du to a transaction timeout or an internal error. This purpose of this counter is mainly for debugging. Transaction abort should not occur in normal operation.

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

clr	Clear transaction abort counter value	
WO/AutoClr	This write autoclear field reset the counter value to 0.	
Possible Values:	0x0	No effect
	0x1	clr the counter value to 0

value (30:0)	Counter value	
RO		
Possible Values:	Any Value	

Section: spi

Address Range: [0x0E0 - 0x0EC]

SPIREGIN SPI Register In

Address: section "spi" base address + 0x000

31	30	29	28	27	26	25	24
Reserved						SPI_OLD_ENABLE	SPI_ENABLE
23	22	21	20	19	18	17	16
Reserved	SPIRW	SPICMDDONE	Reserved		SPISEL	Reserved	SPITXST
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
SPIDATAW(7:0)							

SPI_OLD_ENABLE	
STATIC	This bit is a placeholder for the SPI_ENABLE in older version of the code. It is used both to define a field position in the .h file generated and to mark the bit as reserved in the register file to garantee that the bit will not be re-used in the future.
Value at Reset:	0x0

SPI_ENABLE	SPI ENABLE	
RW	This bit enables the Output enable of the pin of the FPGA. This is needed to put the SPI interface in hi-Z when not using it. Note that this bit has been moved from bit 24 to bit 25 so the existing software will not be compatible with new hardware and there will be no SPI transaction if old software is run over new hardware.	
Value at Reset:	0x0	
Possible Values:	0x0	The SPI interface is disabled
	0x1	The SPI interface is enabled

SPIRW	SPI Read Write	
RW	Specify the SPI transfer type (read or write access).	
Value at Reset:	0x0	
Possible Values:	0x0	Write Access
	0x1	Read Access

SPICMDDONE	SPI CoMmaD DONE
RW	Specify the last transaction for an SPI command sequence.
Value at Reset:	0x0

SPISEL	SPI active channel SELection
RW	Selects the active SPI x channel.
Value at Reset:	0x0

SPITXST	SPI SPITXST Transfer STart
WO/AutoClr	Start an SPI transaction when 1 is written

SPIDATAW (7:0)	SPI Data byte to write
RW	This is the data byte to be written.
Value at Reset:	0x0

SPIREGOUT SPI Register Out

Address: section "spi" base address + 0x008

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

SPI_WB_CAP	SPI Write Burst CAPable	
RO	This register informs if the SPI core is able to write burst of 256 bytes to the SPI device (Write page), without requireing register polling between command, adress and data bytes in the write page command.	
Possible Values:	0x0	This fpga can't do write burst
	0x1	This fpga is capable of doing write burst

SPIWRTD	SPI Write or Read Transfer Done	
RO	Specify if there is a transfer in progress.	
Possible Values:	0x0	Transfer in progress
	0x1	No transfer in progress

SPIDATARD (7:0)	SPI DATA Read byte OUTput
RO	This is the data read byte from the SPI

Section: arbiter

Address Range: [0x0F0 - 0x0F8]

ARBITER_CAPABILITIES

Address: section "arbiter" base address + 0x000

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

AGENT_NB (1:0)	
STATIC	Number of agents
Value at Reset:	0x2

TAG (11:0)	
STATIC	Arbiter TAG : AABiter
Value at Reset:	0xAAB

AGENT (1:0)

Address: section "arbiter" base address + 0x004 + (index * 0x4)

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

ACK	master request ACKnoledge	
RO	This field indicates that a master request has been received, and and the resource is ready to be used.	
Possible Values:	0x0	The resource is NOT ready to be used.
	0x1	The resource is ready to be used.

REC	master request RECeived	
RO	This field indicates that a master request has been received.	
Possible Values:	0x0	Master request not received
	0x1	Master request has been received

DONE	transaction DONE	
WO/AutoClr	This field from master requester informs the arbiter that it has finish with the device resource. The arbiter can give the resource to another master requester.	
Possible Values:	0x0	Nothing
	0x1	Master requester transaction done

REQ	REQuest resource	
WO/AutoClr	This field from master requester ask the arbiter for a device resource.	
Possible Values:	0x0	Nothing
	0x1	Ask for a device resource

Section: axi_window (3 : 0)

Address Range: [0x100 - 0x10C]

Section repeated 4 times. axi_window(i) base address located @ 0x100 + (i * 0x10)

ctrl — PCIe Bar 0 start address

Address: section "axi_window" base address + 0x000

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

enable	
RW	
Value at Reset:	0x0

pci_bar0_start PCIe Bar 0 window start offset

Address: section "axi_window" base address + 0x004

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

value (25:0)	
RW (25:2) *RO (1:0)*	
Value at Reset:	0x0

pci_bar0_stop — PCIe Bar 0 window stop offset

Address: section "axi_window" base address + 0x008

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

value (25:0)	
RW (25:2) *RO (1:0)*	
Value at Reset:	0x0

axi_translation — Axi offset translation

Address: section "axi_window" base address + 0x00C

Description:

32 bits window offset in the axi space

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

value (31:0)	
RW (31:2) *RO (1:0)*	
Value at Reset:	0x0

Section: debug

Address Range: [0x200 - 0x210]

input — debug input signals

Address: section "debug" base address + 0x000

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

value (31:0)	
RO	

output

Address: section "debug" base address + 0x004

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

value (31:0)	
RW	
Value at Reset:	0x0

DMA_DEBUG1

Address: section "debug" base address + 0x008

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

ADD_START (31:0)	
RW	First address of the image buffer in host memory
Value at Reset:	0x0

DMA_DEBUG2

Address: section "debug" base address + 0x00C

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

ADD_OVERRUN (31:0)	
RW	Address of the overrun image buffer in host memory
Value at Reset:	0x0

DMA_DEBUG3

Address: section "debug" base address + 0x010

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

DMA_ADD_ERROR	
RO	Non consecutive TLP adress detected : error (1x clk Event)

DMA_OVERRUN	
RO	Overrun detected (1x clk Event)