SPI Controller (SPI)

20.1 Overview

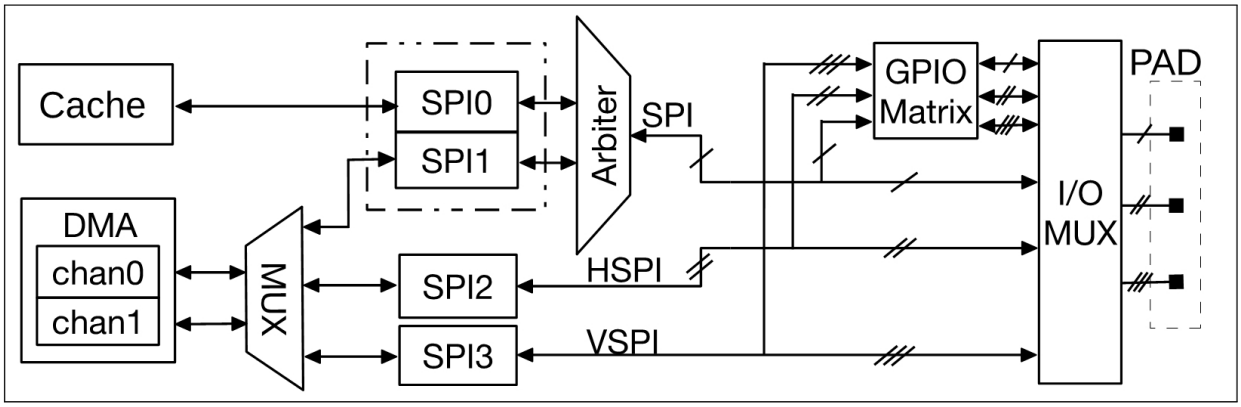


Figure 20.1-1. SPI Architecture

As Figure 20.1-1 shows, ESP32 integrates four SPI controllers which can be used to communicate with external devices that use the SPI protocol. Controller SPI0 is used as a buffer for accessing external memory. Controller SPI1 can be used as a master. Controllers SPI2 and SPI3 can be configured as either a master or a slave. When used as a master, each SPI controller can drive multiple CS signals (CS0~CS2) to activate multiple slaves. Controllers SPI1~SPI3 share two DMA channels.

The SPI signal buses consist of D, Q, CS0-CS2, CLK, WP, and HD signals, as Table 20.1-1 shows. Controllers SPI0 and SPI1 share one signal bus through an arbiter; the signals of the shared bus start with "SPI". Controllers SPI2 and SPI3 use signal buses starting with "HSPI" and "VSPI" respectively. The I/O lines included in the above-mentioned signal buses can be mapped to pins via either the IO_MUX module or the GPIO matrix. (Please refer to Chapter IO_MUX for details.)

The SPI controller supports four-line full-duplex/half-duplex communication (MOSI, MISO, CS, and CLK lines) and three-line half-duplex-only communication (DATA, CS, and CLK lines) in GP-SPI mode. In QSPI mode, an SPI controller accesses the flash or SRAM by using signal buses D, Q, CS0~CS2, CLK, WP, and HD as a four-bit parallel SPI bus. The mapping between SPI bus signals and pin function signals under different communication modes is shown in Table 20.1-1.

Table 20.1-1. Mapping Between SPI Bus Signals and Pin Function Signals

Four-line GP-SPI Full-duplex/half-duplex signal bus	Three-line GP-SPI Half-duplex signal bus	QSPI Signal bus	Pin function signals		
			SPI signal bus	HSPI signal bus	VSPI signal bus
MOSI	DATA	D	SPID	HSPID	VSPID
MISO	-	Q	SPIQ	HSPIQ	VSPIQ

CS	CS	CS	SPICS0	HSPICS0	VSPICS0
CLK	CLK	CLK	SPICLK	HSPICLK	VSPICLK
-	-	WP	SPIWP	HSPIWP	VSPIWP
-	-	HD	SPIHD	HSPIHD	VSPIHD

20.2 SPI Features

General Purpose SPI (GP-SPI)

- Programmable data transfer length, in multiples of 1 byte
- Four-line full-duplex/half-duplex communication and three-line half-duplex communication support
- Master mode and slave mode
- Programmable CPOL and CPHA
- Programmable clock

Parallel QSPI

- Communication format support for specific slave devices such as flash
- Programmable communication format
- Six variations of flash-read operations available
- Automatic shift between flash and SRAM access
- Automatic wait states for flash access

SPI DMA Support

- Support for sending and receiving data using linked lists

SPI Interrupt Hardware

- SPI interrupts
- SPI DMA interrupts

20.3 GP-SPI

The SPI master mode supports four-line full-duplex/half-duplex communication and three-line half-duplex communication. Figure 20.3-1 outlines the connections needed for four-line full-duplex/half-duplex communications.

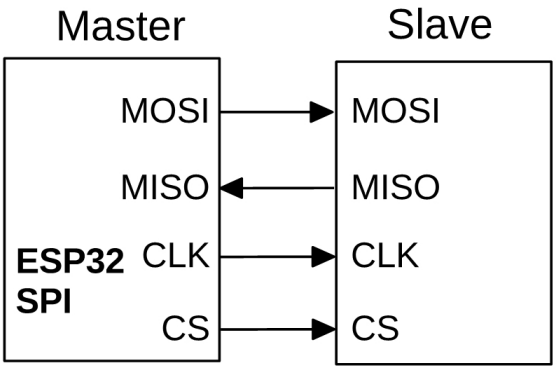


Figure 20.3-1. SPI Master and Slave Full-duplex/Half-duplex Communication

The SPI1~SPI3 controllers can communicate with other slaves as a standard SPI master. SPI2 and SPI3 can be configured as either a master or a slave. Every SPI master can be connected to three slaves at most by default. When not using DMA, the maximum length of data received/sent in one burst is 64 bytes. The data length is in multiples of one byte.

20.3.1 GP-SPI Four-line Full-duplex Communication

When configured to four-line full-duplex mode, the ESP32 SPI can act as either a master or a slave. The length of received and sent data needs to be set by configuring the SPI_MISO_DLEN_REG, SPI_MOSI_DLEN_REG registers for master mode as well as SPI_SLV_RDBUF_DLEN_REG, SPI_SLV_WRBUF_DLEN_REG registers for slave mode. The SPI_DOUTDIN bit and SPI_USR_MOSI bit in register SPI_USER_REG should be configured to enable this communication mode. The SPI_USR bit in register SPI_CMD_REG needs to be configured to initialize a data transfer.

20.3.2 GP-SPI Four-line Half-duplex Communication

When configured to four-line half-duplex mode, the ESP32 SPI can act as either a master or a slave. In this mode, the SPI communication supports flexible communication format as: command + address + dummy phase + received and/or sent data. The format is specified as follows:

1. command: length of 0~16 bits; Master Out Slave In (MOSI).
2. address: length of 0~32/64 bits; Master Out Slave In (MOSI).
3. dummy phase: length of 0~256 SPI clocks.
4. received and/or sent data: length of 0~512 bits (64 bytes); Master Out Slave In (MOSI) or Master In Slave Out (MISO).

The address length is up to 32 bits in GP-SPI master mode and 64 bits in QSPI master mode. The command phase, address phase, dummy phase and received/sent data phase are controlled by bits SPI_USR_COMMAND, SPI_USR_ADDR, SPI_USR_DUMMY, and SPI_USR_MISO/SPI_USR_MOSI respectively in register SPI_USER_REG. A certain phase is enabled only when its corresponding control bit is set to 1. Details can be found in register description. When SPI works as a master, the register can be configured by software as required to determine whether or not to enable a certain phase.

When SPI works as a slave, the communication format must contain command, address, received and/or sent data, among which the command has several options listed in Table 20.3-1. During data transmission or

Table 20.3-1. Command Definitions Supported by GP-SPI Slave in Half-duplex Mode

Command	Description
0x1	Received by slave; writes data sent by the master into the slave status register via MOSI.
0x2	Received by slave; writes data sent by the master into the slave data buffer via MOSI.
0x3	Sent by slave; sends data in the slave buffer to master via MISO.
0x4	Sent by slave; sends data in the slave status register to master via MISO.
0x6	Writes master data on MOSI into data buffer and then sends the date in the slave data buffer to MISO.

reception, the CS signal should keep logic level low. If the CS signal is pulled up during transmission, the internal state of the slave will be reset.

The master can write the slave status register SPI_SLV_WR_STATUS_REG, and decide whether to read data from register SPI_SLV_WR_STATUS_REG or register SPI_RD_STATUS_REG via the SPI_SLV_STATUS_READBACK bit in register SPI_SLAVE1_REG. The SPI master can maintain communication with the slave by reading and writing slave status register, thus realizing complex communication with ease.

The length of received and sent data is controlled by SPI_MISO_DLEN_REG and SPI_MOSI_DLEN_REG in master mode, as well as SPI_SLV_RDBUF_DLEN_REG and SPI_SLV_WRBUF_DLEN_REG in slave mode. A reception or transmission of data is controlled by bit SPI_USR_MOSI or SPI_USR_MISO in SPI_USER_REG. The SPI_USR bit in register SPI_CMD_REG needs to be configured to initialize a data transfer.

20.3.3 GP-SPI Three-line Half-duplex Communication

The three-line half-duplex communication differs from four-line half-duplex communication in that the reception and transmission shares one signal bus and that the communication format must contain command, address, received and/or sent data. Software can enable three-line half-duplex communication by configuring SPI_SIO bit in SPI_USER_REG register.

Note:

- In half-duplex communication, the order of command, address, received and/or sent data in the communication format should be followed strictly.
- In half-duplex communication, communication formats "command + address + received data + sent data" and "received data + sent data" are not applicable to DMA.
- When ESP32 SPI acts as a slave, the master CS should be active at least one SPI clock period before a read/write process is initiated, and should be inactive at least one SPI clock period after the read/write process is completed.

20.3.4 GP-SPI Data Buffer

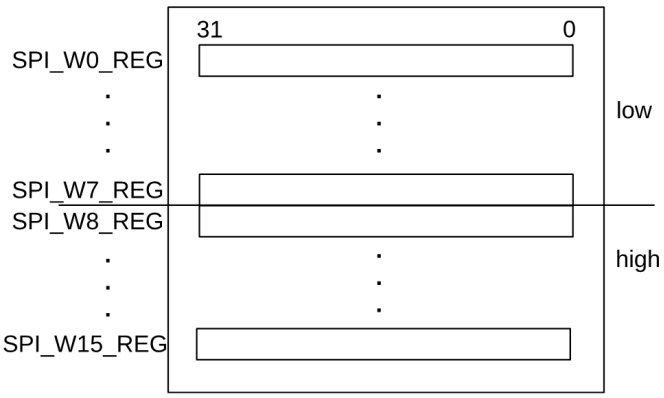


Figure 20.3-2. SPI Data Buffer

ESP32 SPI has 16 × 32 bits of data buffer to buffer data-send and data-receive operations. As is shown in Figure 20.3-2, received data is written from the low byte of SPI_W0_REG by default and the writing ends with SPI_W15_REG. If the data length is over 64 bytes, the extra part will be written from SPI_W0_REG.

Data buffer blocks SPI_W0_REG ~ SPI_W7_REG and SPI_W8_REG ~ SPI_W15_REG data correspond to the lower part and the higher part respectively. They can be used separately, and are controlled by the SPI_USR_MOSI
_HIGHPART bit and the SPI_USR_MISO_HIGHPART bit in register SPI_USER_REG. For example, if SPI is configured as a master, when SPI_USR_MOSI_HIGHPART = 1, SPI_W8_REG ~ SPI_W15_REG are used as buffer for sending data; when SPI_USR_MISO_HIGHPART = 1, SPI_W8_REG ~ SPI_W15_REG are used as buffer for receiving data. If SPI acts as a slave, when SPI_USR_MOSI_HIGHPART = 1, SPI_W8_REG ~ SPI_W15_REG are used as buffer for receiving data; when SPI_USR_MISO_HIGHPART = 1, SPI_W8_REG ~ SPI_W15_REG are used as buffer for sending data.

20.4 GP-SPI Clock Control

The maximum output clock frequency of ESP32 GP-SPI master is $f_{\text{apb}}/2$, and the maximum input clock frequency of the ESP32 GP-SPI slave is $f_{\text{apb}}/8$. The master can derive other clock frequencies via frequency division.

$$f_{\text{spi}} = \frac{f_{\text{apb}}}{(\text{SPI\_CLKCNT\_N+1})(\text{SPI\_CLKDIV\_PRE+1})}$$

SPI_CLKCNT_N and SPI_CLKDIV_PRE are two bits of register SPI_CLOCK_REG (Please refer to 20.7 Register Description for details). SPI_CLKCNT_H $= \lfloor \frac{\text{SPI\_CLKCNT\_N+1}}{2} - 1 \rfloor$, SPI_CLKCNT_N=SPI_CLKCNT_L. When the SPI_CLK_EQU_SYSCLK bit in register SPI_CLOCK_REG is set to 1, and the other bits are set to 0, SPI output clock frequency is f_{apb}. For other clock frequencies, SPI_CLK_EQU_SYSCLK needs to be 0. In slave mode, SPI_CLKCNT_N, SPI_CLKCNT_L, SPI_CLKCNT_H and SPI_CLKDIV_PRE should all be 0.

20.4.1 GP-SPI Clock Polarity (CPOL) and Clock Phase (CPHA)

The clock polarity and clock phase of ESP32 SPI are controlled by SPI_CK_IDLE_EDGE bit in register SPI_PIN_REG, SPI_CK_OUT_EDGE bit and SPI_CK_I_EDGE bit in register SPI_USER_REG, as well as

Table 20.4-1. Clock Polarity and Phase, and Corresponding SPI Register Values for SPI Master

Registers	mode0	mode1	mode2	mode3
SPI_CK_IDLE_EDGE	0	0	1	1
SPI_CK_OUT_EDGE	0	1	1	0
SPI_MISO_DELAY_MODE	2(0)	1(0)	1(0)	2(0)
SPI_MISO_DELAY_NUM	0	0	0	0
SPI_MOSI_DELAY_MODE	0	0	0	0
SPI_MOSI_DELAY_NUM	0	0	0	0

SPI_MISO_DELAY_MODE[1:0] bit, SPI_MISO_DELAY_NUM[2:0] bit, SPI_MOSI_DELAY_MODE[1:0] bit, SPI_MOSI_DELAY_MUM[2:0] bit in register SPI_CTRL2_REG. Table 20.4-1 and Table 20.4-2 show the clock polarity and phase as well as the corresponding register values for ESP32 SPI master and slave, respectively. Note that for mode0 and mode2 in Table 20.4-2, the registers are configured differently in non-DMA mode and DMA mode, and that the SPI slave data is output in advance in DMA mode.

Table 20.4-2. Clock Polarity and Phase, and Corresponding SPI Register Values for SPI Slave

Registers	mode0		mode1	mode2		mode3
	Non-DMA	DMA		Non-DMA	DMA	
SPI_CK_IDLE_EDGE	1	0	1	0	1	0
SPI_CK_I_EDGE	0	1	1	1	0	0
SPI_MISO_DELAY_MODE	0	0	2	0	0	1
SPI_MISO_DELAY_NUM	0	2	0	0	2	0
SPI_MOSI_DELAY_MODE	2	0	0	1	0	0
SPI_MOSI_DELAY_NUM	2	3	0	2	3	0

1. mode0 means CPOL=0, CPHA=0. When SPI is idle, the clock output is logic low; data changes on the falling edge of the SPI clock and is sampled on the rising edge;
2. mode1 means CPOL=0, CPHA=1. When SPI is idle, the clock output is logic low; data changes on the rising edge of the SPI clock and is sampled on the falling edge;

3. mode2 means when CPOL=1, CPHA=0. When SPI is idle, the clock output is logic high; data changes on the rising edge of the SPI clock and is sampled on the falling edge;
4. mode3 means when CPOL=1, CPHA=1. When SPI is idle, the clock output is logic high; data changes on the falling edge of the SPI clock and is sampled on the rising edge.

20.4.2 GP-SPI Timing

The data signals of ESP32 GP-SPI can be mapped to physical pins either via IO_MUX or via IO_MUX and GPIO matrix. Input signals will be delayed by two clk_{apb} clock cycles when they pass through the matrix. Output signals will not be delayed.

When GP-SPI is used as master and the data signals are not received by the SPI controller via GPIO matrix, if GP-SPI output clock frequency is $clk_{\text{apb}}/2$, register SPI_MISO_DELAY_MODE should be set to 0 when configuring the clock polarity. If GP-SPI output clock frequency is not higher than $clk_{\text{apb}}/4$, register SPI_MISO_DELAY_MODE can be set to the corresponding value in Table 20.4-1 when configuring the clock polarity.

When GP-SPI is used in master mode and the data signals enter the SPI controller via the GPIO matrix:

1. If GP-SPI output clock frequency is $clk_{\text{apb}}/2$, register SPI_MISO_DELAY_MODE should be set to 0 and the dummy phase should be enabled (SPI_USR_DUMMY = 1) for one clk_{spi} clock cycle (SPI_USR_DUMMY_CYC
   LELEN = 0) when configuring the clock polarity;
2. If GP-SPI output clock frequency is $clk_{\text{apb}}/4$, register SPI_MISO_DELAY_MODE should be set to 0 when configuring the clock polarity;
3. If GP-SPI output clock frequency is not higher than $clk_{\text{apb}}/8$, register SPI_MISO_DELAY_MODE can be set to the corresponding value in Table 20.4-1 when configuring the clock polarity.

When GP-SPI is used in slave mode, the clock signal and the data signals should be routed to the SPI controller via the same path, i.e., neither the clock signal nor the data signals passes through GPIO matrix, or both of them pass through GPIO matrix. This is important in ensuring that the signals are not delayed by different time periods before they reach the SPI hardware.

Assume that t_{spi}, t_{pre} and t_{v} in Figure 20.4-1 denote SPI clock period, how far ahead data output is, and data output delay time, respectively. Assume the SPI slave's main clock period is t_{apb}. For non-DMA mode0, SPI slave data output is delayed by t_{v}:

- $t_{\text{v}} < 3.5 * t_{\text{apb}}$, if CLK does not pass through GPIO matrix;
- $t_{\text{v}} < 5.5 * t_{\text{apb}}$, if CLK passes through GPIO matrix.

In DMA mode1 and mode3, SPI slave data output is delayed by the same period of time as in non-DMA mode. However, for mode0 and mode2, SPI slave data is output earlier by t_{pre}:

- $t_{\text{pre}} < (t_{\text{spi}}/2 - 5.5 * t_{\text{apb}})$, if CLK does not pass through GPIO matrix;
- $t_{\text{pre}} < (t_{\text{spi}}/2 - 7.5 * t_{\text{apb}})$, if CLK passes through GPIO matrix.

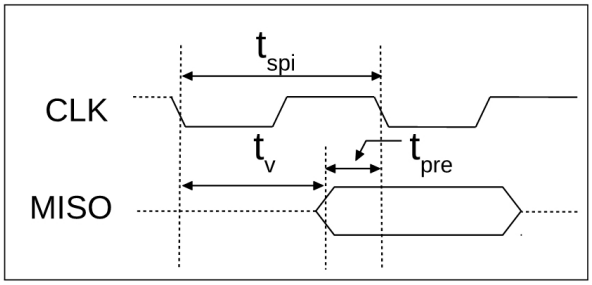


Figure 20.4-1. GP-SPI Slave Data Output

To conclude, if signals do not pass through GPIO matrix, the SPI slave clock frequency is up to $f_{\text{apb}}/8$; if signals pass through GPIO matrix, the SPI slave clock frequency is up to $f_{\text{apb}}/12$. Note that $(t_{\text{spi}}/2-t_{\text{pre}})$ represents data output hold time for SPI slave in mode0 and mode2.

20.5 Parallel QSPI

ESP32 SPI controllers support SPI bus memory devices (such as flash and SRAM). The hardware connection between the SPI pins and the memories is shown by Figure 20.5-1.

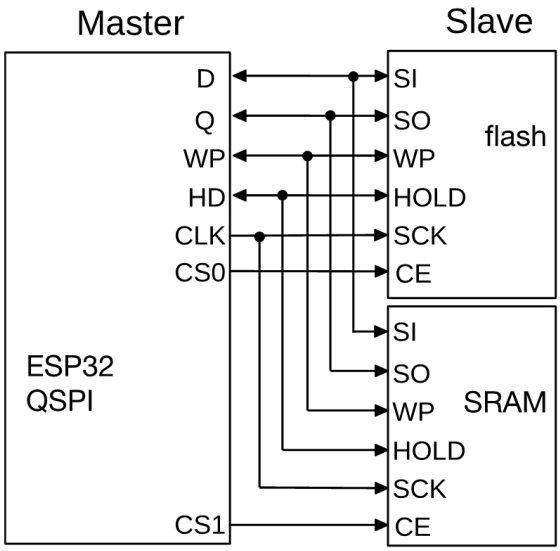


Figure 20.5-1. Parallel QSPI

SPI1, SPI2 and SPI3 controllers can also be configured as QSPI master to connect to external memory. The maximum output clock frequency of the SPI memory interface is f_{apb}, with the same clock configuration as that of the GP-SPI master.

20.5.1 Communication Format of Parallel QSPI

To support communication with special slave devices, ESP32 QSPI implements a specifically designed communication protocol. The communication format of ESP32 QSPI master is the same as that of GP-SPI four-line half-duplex communication, except that in address phase and data phase, software can configure registers to enable two-line or four-line transmission. Figure 20.5-2 shows a QSPI communication mode with four-line transmission in address phase and data phase.

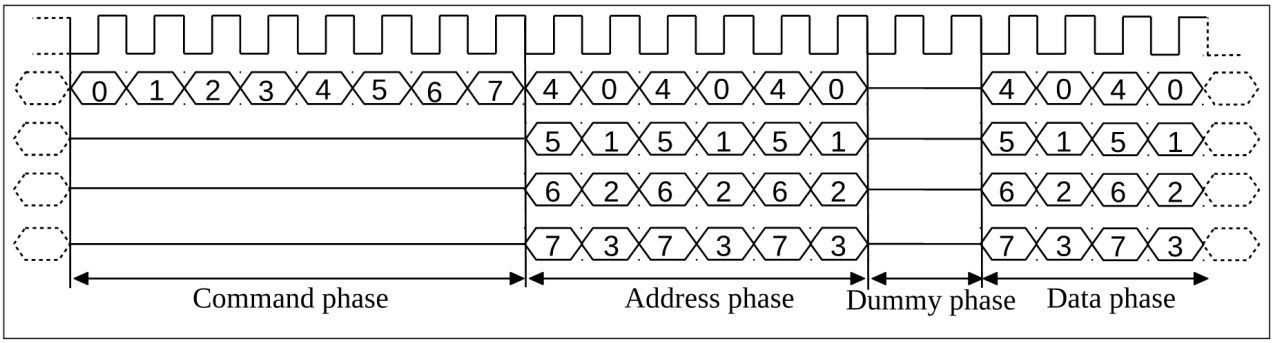


Figure 20.5-2. Communication Format of Parallel QSPI

ESP32 QSPI supports flash-read operation in one-line, two-line, and four-line modes. When working as a QSPI master, the command phase, address phase, dummy phase and data phase can be configured as needed, as flexible as in GP-SPI mode.

Note that GPI-SPI full-duplex mode does not support dummy phase.

20.6 GP-SPI Interrupt Hardware

ESP32 SPI generates two types of interrupts. One is the SPI interrupt and the other is the SPI DMA interrupt.

ESP32 SPI reckons the completion of send- and/or receive-operations as the completion of one operation from the controller and generates one interrupt. When ESP32 SPI is configured to slave mode, the slave will generate read/write status registers and read/write buffer data interrupts according to different operations.

20.6.1 SPI Interrupts

The SPI_*_INTEN bits in the SPI_SLAVE_REG register can be set to enable SPI interrupts. When an SPI interrupt happens, the interrupt flag in the corresponding SPI_*_DONE register will get set. This flag is writable, and an interrupt can be cleared by setting the bit to zero.

- SPI_TRANS_DONE_INT: Triggered when an SPI operation is done.
- SPI_SLV_WR_STA_INT: Triggered when an SPI slave status write is done.
- SPI_SLV_RD_STA_INT: Triggered when an SPI slave status read is done.
- SPI_SLV_WR_BUF_INT: Triggered when an SPI slave buffer write is done.
- SPI_SLV_RD_BUD_INT: Triggered when an SPI slave buffer read is done.

20.6.2 DMA Interrupts

- SPI_OUT_TOTAL_EOF_INT: Triggered when all linked lists are sent.
- SPI_OUT_EOF_INT: Triggered when one linked list is sent.
- SPI_OUT_DONE_INT: Triggered when the last linked list item has zero length.
- SPI_IN_SUC_EOF_INT: Triggered when all linked lists are received.

- SPI_IN_ERR_EOF_INT: Triggered when there is an error receiving linked lists.
- SPI_IN_DONE_INT: Triggered when the last received linked list had a length of 0.
- SPI_INLINK_DSCR_ERROR_INT: Triggered when the received linked list is invalid.
- SPI_OUTLINK_DSCR_ERROR_INT: Triggered when the linked list to be sent is invalid.
- SPI_INLINK_DSCR_EMPTY_INT: Triggered when no valid linked list is available.

20.7 Register Summary

The addresses in this section are relative to the SPI base address provided in Table 3.3-6 in Chapter 3 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	SPI0	SPI1	SPI2	SPI3	Acc
Control and configuration registers						
SPI_CTRL_REG	Bit order and QIO/DIO/QOUT/DOUT mode settings	3FF43008	3FF42008	3FF64008	3FF65008	R/W
SPI_CTRL2_REG	Timing configuration	3FF43014	3FF42014	3FF64014	3FF65014	R/W
SPI_CLOCK_REG	Clock configuration	3FF43018	3FF42018	3FF64018	3FF65018	R/W
SPI_PIN_REG	Polarity and CS configuration	3FF43034	3FF42034	3FF64034	3FF65034	R/W
Slave mode configuration registers						
SPI_SLAVE_REG	Slave mode configuration and interrupt status	3FF43038	3FF42038	3FF64038	3FF65038	R/W
SPI_SLAVE1_REG	Slave data bit lengths	3FF4303C	3FF4203C	3FF6403C	3FF6503C	R/W
SPI_SLAVE2_REG	Dummy cycle length configuration	3FF43040	3FF42040	3FF64040	3FF65040	R/W
SPI_SLV_WR_STATUS_REG	Slave status/Part of lower master address	3FF43030	3FF42030	3FF64030	3FF65030	R/W
SPI_SLV_WRBUF_DLEN_REG	Write-buffer operation length	3FF43048	3FF42048	3FF64048	3FF65048	R/W
SPI_SLV_RDBUF_DLEN_REG	Read-buffer operation length	3FF4304C	3FF4204C	3FF6404C	3FF6504C	R/W
SPI_SLV_RD_BIT_REG	Read data operation length	3FF43064	3FF42064	3FF64064	3FF65064	R/W
User-defined command mode registers						
SPI_CMD_REG	Start user-defined command	3FF43000	3FF42000	3FF64000	3FF65000	R/W
SPI_ADDR_REG	Address data	3FF43004	3FF42004	3FF64004	3FF65004	R/W

Name	Description	SPI0	SPI1	SPI2	SPI3	Acc
SPI_USER_REG	User defined command configuration	3FF4301C	3FF4201C	3FF6401C	3FF6501C	R/W
SPI_USER1_REG	Address and dummy cycle configuration	3FF43020	3FF42020	3FF64020	3FF65020	R/W
SPI_USER2_REG	Command length and value configuration	3FF43024	3FF42024	3FF64024	3FF65024	R/W
SPI_MOSI_DLEN_REG	MOSI length	3FF43028	3FF42028	3FF64028	3FF65028	R/W
SPI_W0_REG	SPI data register 0	3FF43080	3FF42080	3FF64080	3FF65080	R/W
SPI_W1_REG	SPI data register 1	3FF43084	3FF42084	3FF64084	3FF65084	R/W
SPI_W2_REG	SPI data register 2	3FF43088	3FF42088	3FF64088	3FF65088	R/W
SPI_W3_REG	SPI data register 3	3FF4308C	3FF4208C	3FF6408C	3FF6508C	R/W
SPI_W4_REG	SPI data register 4	3FF43090	3FF42090	3FF64090	3FF65090	R/W
SPI_W5_REG	SPI data register 5	3FF43094	3FF42094	3FF64094	3FF65094	R/W
SPI_W6_REG	SPI data register 6	3FF43098	3FF42098	3FF64098	3FF65098	R/W
SPI_W7_REG	SPI data register 7	3FF4309C	3FF4209C	3FF6409C	3FF6509C	R/W
SPI_W8_REG	SPI data register 8	3FF430A0	3FF420A0	3FF640A0	3FF650A0	R/W
SPI_W9_REG	SPI data register 9	3FF430A4	3FF420A4	3FF640A4	3FF650A4	R/W
SPI_W10_REG	SPI data register 10	3FF430A8	3FF420A8	3FF640A8	3FF650A8	R/W
SPI_W11_REG	SPI data register 11	3FF430AC	3FF420AC	3FF640AC	3FF650AC	R/W
SPI_W12_REG	SPI data register 12	3FF430B0	3FF420B0	3FF640B0	3FF650B0	R/W
SPI_W13_REG	SPI data register 13	3FF430B4	3FF420B4	3FF640B4	3FF650B4	R/W
SPI_W14_REG	SPI data register 14	3FF430B8	3FF420B8	3FF640B8	3FF650B8	R/W
SPI_W15_REG	SPI data register 15	3FF430BC	3FF420BC	3FF640BC	3FF650BC	R/W
DMA configuration registers						
SPI_DMA_CONF_REG	DMA configuration register	3FF43100	3FF42100	3FF64100	3FF65100	R/W
SPI_DMA_OUT_LINK_REG	DMA outlink address and configuration	3FF43104	3FF42104	3FF64104	3FF65104	R/W
SPI_DMA_IN_LINK_REG	DMA inlink address and configuration	3FF43108	3FF42108	3FF64108	3FF65108	R/W
SPI_DMA_STATUS_REG	DMA status	3FF4310C	3FF4210C	3FF6410C	3FF6510C	RO
SPI_IN_ERR_EOF_DES_ADDR_REG	Descriptor address where an error occurs	3FF43120	3FF42120	3FF64120	3FF65120	RO
SPI_IN_SUC_EOF_DES_ADDR_REG	Descriptor address where EOF occurs	3FF43124	3FF42124	3FF64124	3FF65124	RO
SPI_INLINK_DSCR_REG	Current descriptor pointer	3FF43128	3FF42128	3FF64128	3FF65128	RO
SPI_INLINK_DSCR_BF0_REG	Next descriptor data pointer	3FF4312C	3FF4212C	3FF6412C	3FF6512C	RO
SPI_INLINK_DSCR_BF1_REG	Current descriptor data pointer	3FF43130	3FF42130	3FF64130	3FF65130	RO

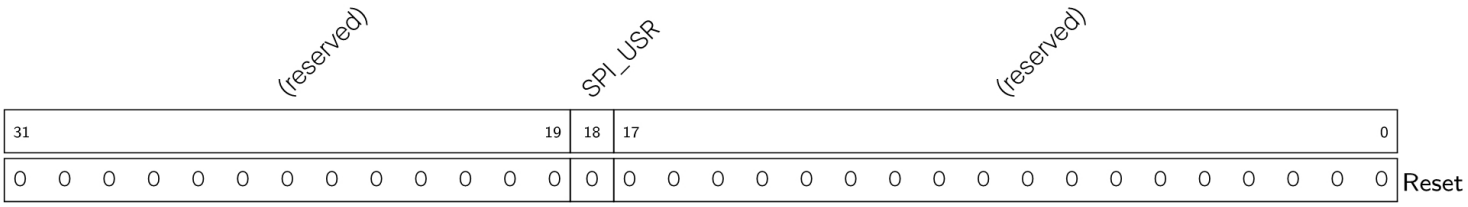
Name	Description	SPI0	SPI1	SPI2	SPI3	Acc
SPI_OUT_EOF_BFR_DES_ADDR_REG	Relative buffer address where EOF occurs	3FF43134	3FF42134	3FF64134	3FF65134	RO
SPI_OUT_EOF_DES_ADDR_REG	Descriptor address where EOF occurs	3FF43138	3FF42138	3FF64138	3FF65138	RO
SPI_OUTLINK_DSCR_REG	Current descriptor pointer	3FF4313C	3FF4213C	3FF6413C	3FF6513C	RO
SPI_OUTLINK_DSCR_BF0_REG	Next descriptor data pointer	3FF43140	3FF42140	3FF64140	3FF65140	RO
SPI_OUTLINK_DSCR_BF1_REG	Current descriptor data pointer	3FF43144	3FF42144	3FF64144	3FF65144	RO
SPI_DMA_RSTATUS_REG	DMA memory read status	3FF43148	3FF42148	3FF64148	3FF65148	RO
SPI_DMA_TSTATUS_REG	DMA memory write status	3FF4314C	3FF4214C	3FF6414C	3FF6514C	RO
DMA interrupt registers						
SPI_DMA_INT_RAW_REG	Raw interrupt status	3FF43114	3FF42114	3FF64114	3FF65114	RO
SPI_DMA_INT_ST_REG	Masked interrupt status	3FF43118	3FF42118	3FF64118	3FF65118	RO
SPI_DMA_INT_ENA_REG	Interrupt enable bits	3FF43110	3FF42110	3FF64110	3FF65110	R/W
SPI_DMA_INT_CLR_REG	Interrupt clear bits	3FF4311C	3FF4211C	3FF6411C	3FF6511C	R/W

20.8 Registers

The addresses in this section are relative to the SPI base address provided in Table 3.3-6 in Chapter 3 *System and Memory*. The absolute register addresses are listed in Section 20.7 *Register Summary*.

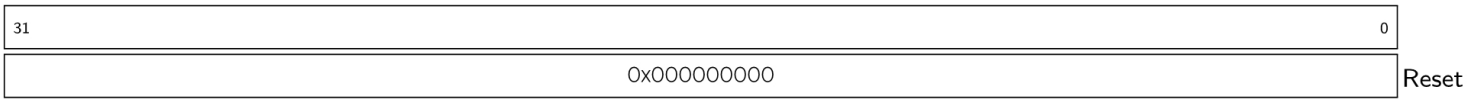
For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 20.1. SPI_CMD_REG (0x0)



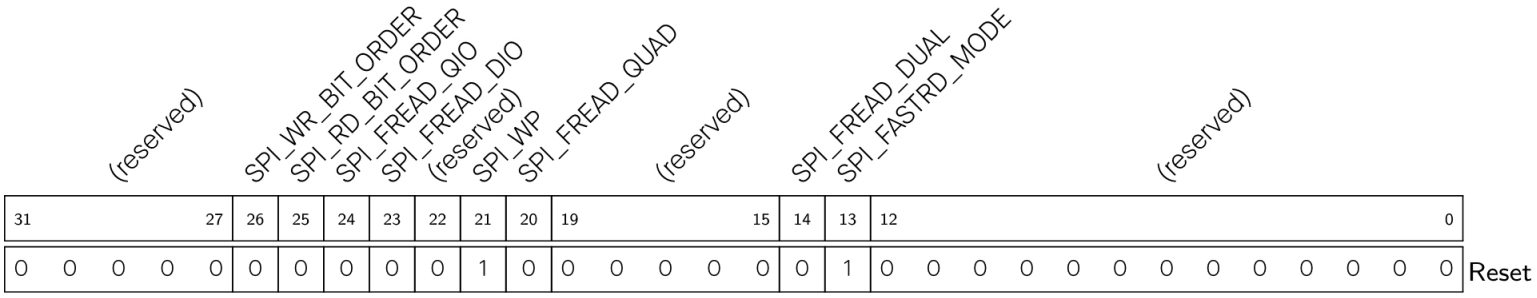
SPI_USR An SPI operation will be triggered when this bit is set. The bit will be cleared once the operation is done. (R/W)

Register 20.2. SPI_ADDR_REG (0x4)



SPI_ADDR_REG It stores the transmitting address when master is in half-duplex mode or QSPI mode. If the address length is bigger than 32 bits, this register stores the higher 32 bits of address value, SPI_SLV_WR_STATUS_REG stores the rest lower part of address value. If the address length is smaller than 33 bits, this register stores all the address value. The register is in valid only when SPI_USR_ADDR bit is set to 1. (R/W)

Register 20.3. SPI_CTRL_REG (0x8)



SPI_WR_BIT_ORDER This bit determines the bit order for command, address and data in transmitted signal. 1: sends LSB first; 0: sends MSB first. (R/W)

SPI_RD_BIT_ORDER This bit determines the bit order for received data in received signal. 1: receives LSB first; 0: receives MSB first. (R/W)

SPI_FREAD_QIO This bit is used to enable four-line address writes and data reads in QSPI mode. (R/W)

SPI_FREAD_DIO This bit is used to enable two-line address writes and data reads in QSPI mode. (R/W)

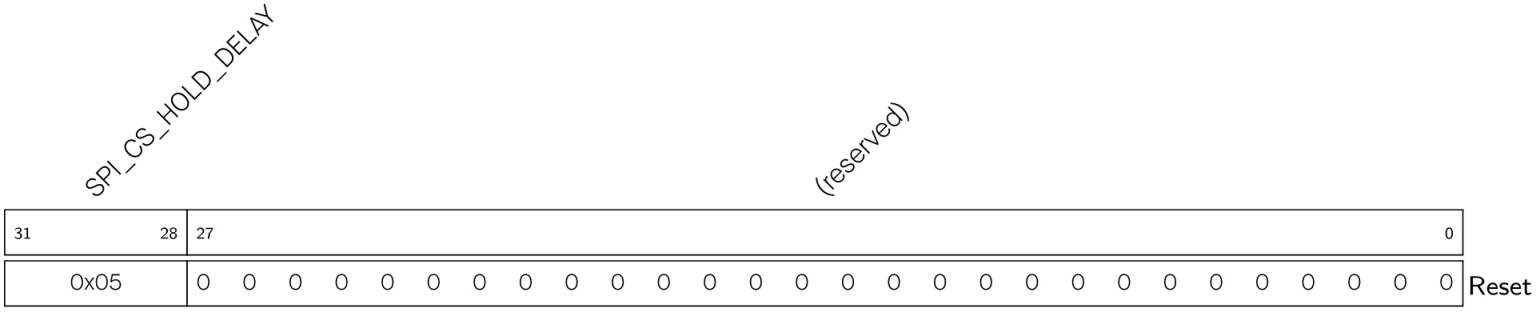
SPI_WP This bit determines the write-protection signal output when SPI is idle in QSPI mode. 1: output high; 0: output low. (R/W)

SPI_FREAD_QUAD This bit is used to enable four-line data reads in QSPI mode. (R/W)

SPI_FREAD_DUAL This bit is used to enable two-line data reads in QSPI mode. (R/W)

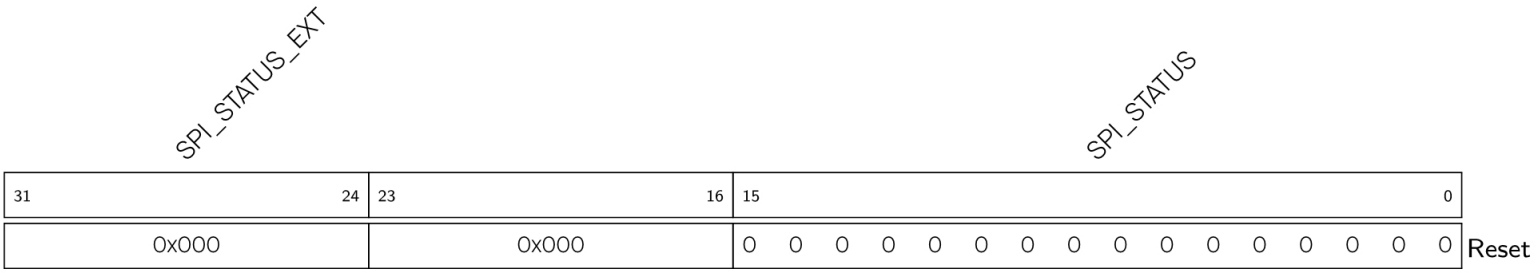
SPI_FASTRD_MODE Reserved.

Register 20.4. SPI_CTRL1_REG (0xC)



SPI_CS_HOLD_DELAY Reserved.

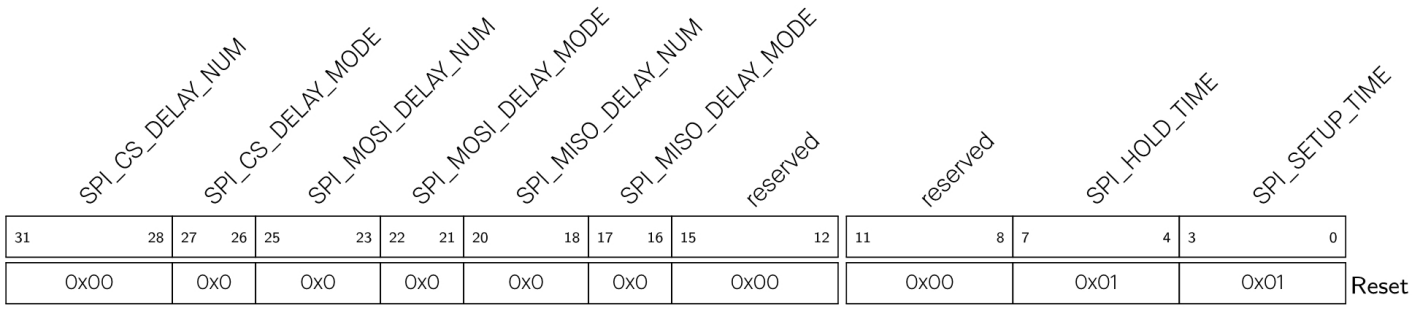
Register 20.5. SPI_RD_STATUS_REG (0x10)



SPI_STATUS_EXT Reserved.

SPI_STATUS Reserved.

Register 20.6. SPI_CTRL2_REG (0x14)



SPI_CS_DELAY_NUM Reserved.

SPI_CS_DELAY_MODE Reserved.

SPI_MOSI_DELAY_NUM It is used to configure the number of system clock cycles by which the MOSI signals are delayed. (R/W)

SPI_MOSI_DELAY_MODE This register field determines the way the MOSI signals are delayed by SPI clock. (R/W)

After being delayed by SPI_MOSI_DELAY_NUM system clocks, the MOSI signals will then be delayed by the configuration of SPI_MOSI_DELAY_MODE, specifically:

0: no delay.

1: if SPI_CK_OUT_EDGE or SPI_CK_I_EDGE is set, the MOSI signals are delayed by half a cycle, otherwise they are delayed by one cycle.

2: if SPI_CK_OUT_EDGE or SPI_CK_I_EDGE is set, the MOSI signals are delayed by one cycle, otherwise they are delayed by half a cycle.

3: the MOSI signals are delayed one cycle.

SPI_MISO_DELAY_NUM It is used to configure the number of system clock cycles by which the MISO signals are delayed. (R/W)

SPI_MISO_DELAY_MODE This register field determines the way MISO signals are delayed by SPI clock. (R/W)

After being delayed by SPI_MISO_DELAY_NUM system clock, the MISO signals will then be delayed by the configuration of SPI_MISO_DELAY_MODE, specifically:

0: no delay.

1: if SPI_CK_OUT_EDGE or SPI_CK_I_EDGE is set, the MISO signals are delayed by half a cycle, otherwise they are delayed by one cycle.

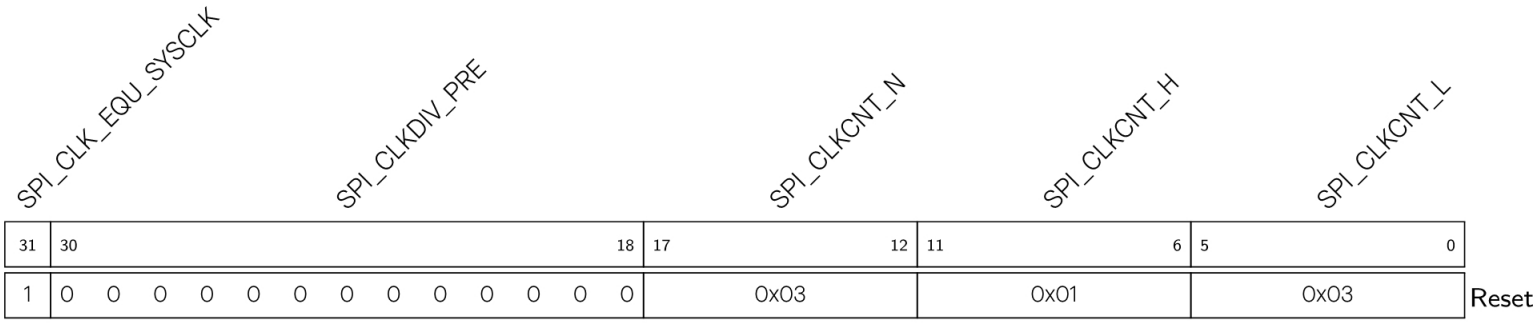
2: if SPI_CK_OUT_EDGE or SPI_CK_I_EDGE is set, the MISO signals are delayed by one cycle, otherwise they are delayed by half a cycle.

3: the MISO signals are delayed by one cycle.

SPI_HOLD_TIME The number of SPI clock cycles by which CS pin signals are delayed. It is only valid when SPI_CS_HOLD is set to 1. (R/W)

SPI_SETUP_TIME It is to configure the time between the CS signal active edge and the first SPI clock edge. It is only valid in half-duplex mode or QSPI mode and when SPI_CS_SETUP is set to 1. (R/W)

Register 20.7. SPI_CLOCK_REG (0x18)



SPI_CLK_EQU_SYSCLK In master mode, when this bit is set to 1, SPI output clock is equal to system clock; when set to 0, SPI output clock is divided from system clock. In slave mode, it should be set to 0. (R/W)

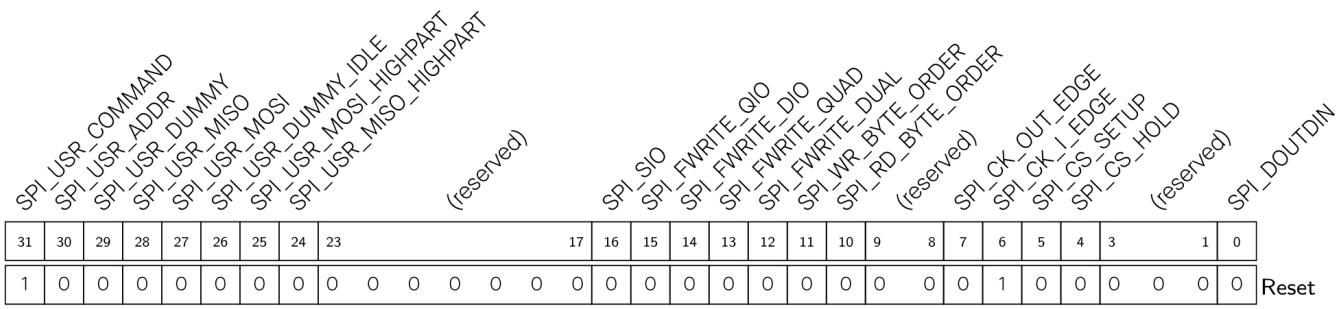
SPI_CLKDIV_PRE In master mode, it is used to configure the pre-divider value for SPI output clock. It is only valid when SPI_CLK_EQU_SYSCLK is 0. In slave mode, it should be set to 0. (R/W)

SPI_CLKCNT_N In master mode, it is used to configure the divider for SPI output clock. It is only valid when SPI_CLK_EQU_SYSCLK is 0. In slave mode, it should be set to 0. (R/W)

SPI_CLKCNT_H In master mode, SPI_CLKCNT_H $= \lfloor \frac{\text{SPI\_CLKCNT\_N}+1}{2} - 1 \rfloor$. It is only valid when SPI_CLK_EQU_SYSCLK is 0. In slave mode, it should be set to 0. (R/W)

SPI_CLKCNT_L In master mode, it is equal to SPI_CLKCNT_N. It is only valid when SPI_CLK_EQU_SYSCLK is 0. In slave mode, it should be set to 0. (R/W)

Register 20.8. SPI_USER_REG (0x1C)



SPI_USR_COMMAND This bit enables the command phase of an SPI operation in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_ADDR This bit enables the address phase of an SPI operation in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_DUMMY This bit enables the dummy phase of an SPI operation in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_MISO This bit enables the read-data phase of an SPI operation in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_MOSI This bit enables the write-data phase of an SPI operation in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_DUMMY_IDLE The SPI clock signal is disabled in the dummy phase when the bit is set in SPI half-duplex mode and QSPI mode. (R/W)

SPI_USR_MOSI_HIGHPART If set, MOSI data is stored in SPI_W8 ~ SPI_W15 of the SPI buffer. (R/W)

SPI_USR_MISO_HIGHPART If set, MISO data is stored in SPI_W8 ~ SPI_W15 of the SPI buffer. (R/W)

SPI_SIO Set this bit to enable three-line half-duplex communication. (R/W)

SPI_FWRITE_QIO Reserved.

SPI_FWRITE_DIO Reserved.

SPI_FWRITE_QUAD Reserved.

SPI_FWRITE_DUAL Reserved.

SPI_WR_BYTE_ORDER This bit determines the byte order of the command, address and data in transmitted signal. 1: big-endian; 0: little-endian. (R/W)

SPI_RD_BYTE_ORDER This bit determines the byte order of received data in transmitted signal. 1: big-endian; 0: little_endian. (R/W)

SPI_CK_OUT_EDGE This bit, combined with SPI_MOSI_DELAY_MODE, sets the MOSI signal delay mode. It is only valid in master mode. (R/W)

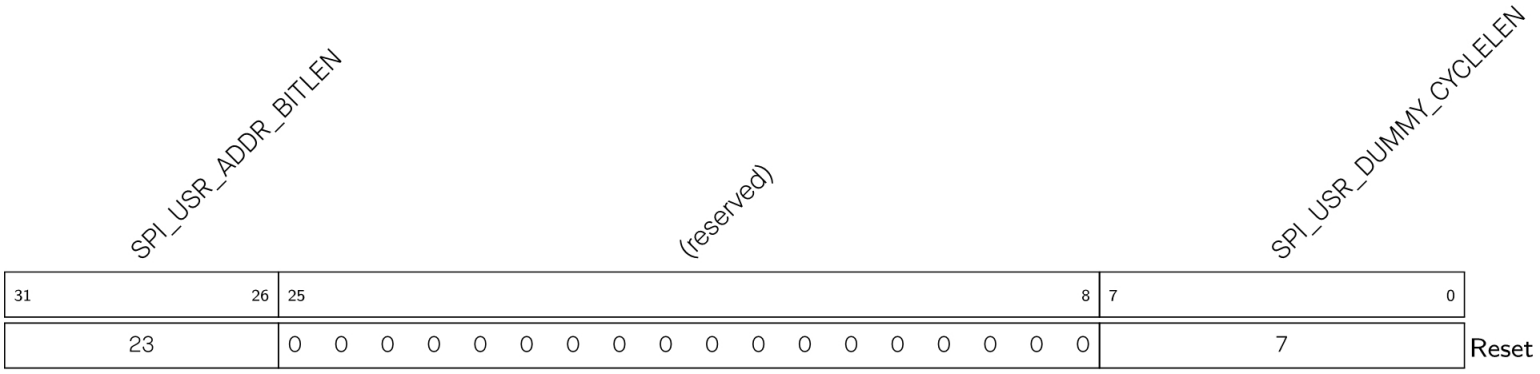
SPI_CK_I_EDGE In slave mode, the bit is the same as SPI_CK_OUT_EDGE in master mode. It is combined with SPI_MISO_DELAY_MODE. It is only valid in slave mode. (R/W)

SPI_CS_SETUP Setting this bit enables a delay between CS active edge and the first clock edge, in multiples of one SPI clock cycle. In full-duplex mode and QSPI mode, setting this bit results in (SPI_SETUP_TIME + 1.5) SPI clock cycles delay. In full-duplex mode, there will be 1.5 SPI clock cycles delay for mode0 and mode2, and 1 SPI clock cycle delay for mode1 and mode3. (R/W)

SPI_CS_HOLD Setting this bit enables a delay between the end of a transmission and CS being inactive, as specified in SPI_HOLD_TIME. (R/W)

SPI_DOUTDIN Set the bit to enable full-duplex communication. (R/W)

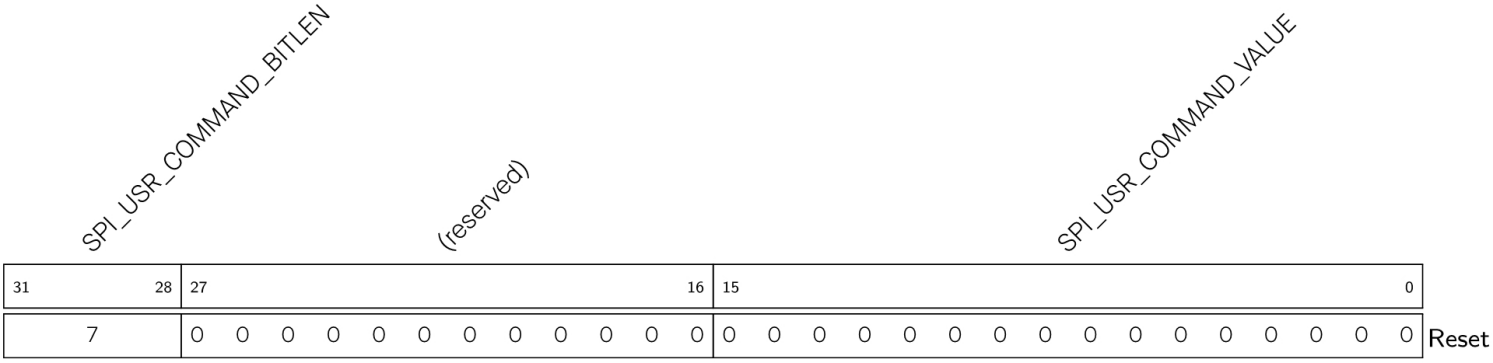
Register 20.9. SPI_USER1_REG (0x20)



SPI_USR_ADDR_BITLEN It indicates the bit length of the transmitted address minus one in half-duplex mode and QSPI mode, in multiples of one bit. It is only valid when SPI_USR_ADDR is set to 1. (RO)

SPI_USR_DUMMY_CYCLELEN It indicates the number of SPI clock cycles for the dummy phase minus one in SPI half-duplex mode and QSPI mode. It is only valid when SPI_USR_DUMMY is set to 1. (R/W)

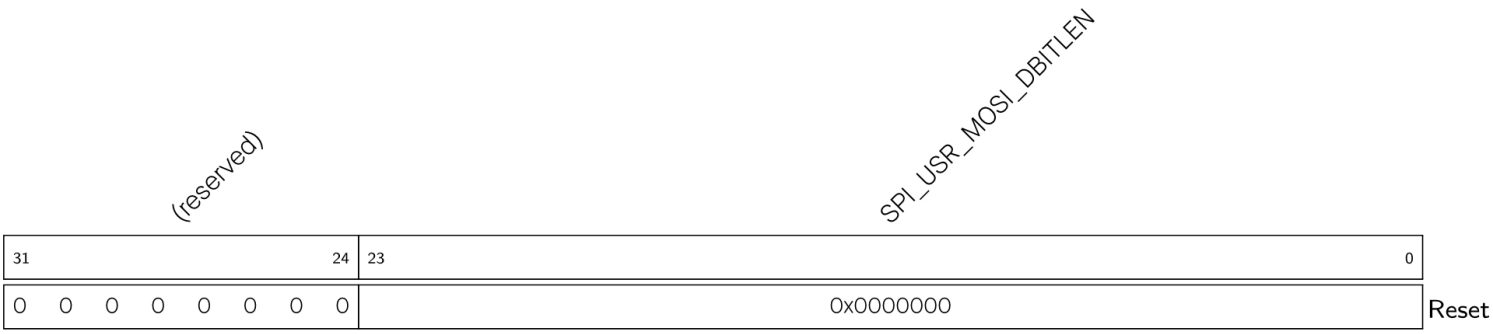
Register 20.10. SPI_USER2_REG (0x24)



SPI_USR_COMMAND_BITLEN It indicates the bit length of the command phase minus one in SPI half-duplex mode and QSPI mode. It is only valid when SPI_USR_COMMAND is set to 1. (R/W)

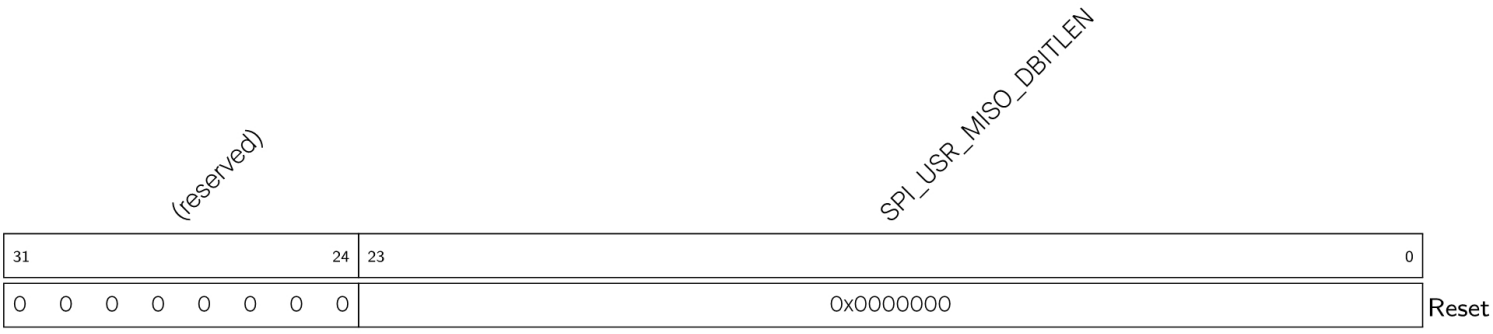
SPI_USR_COMMAND_VALUE It indicates the value of the command to be transmitted in SPI half-duplex mode and QSPI mode. It is only valid when SPI_USR_COMMAND is set to 1. (R/W)

Register 20.11. SPI_MOSI_DLEN_REG (0x28)



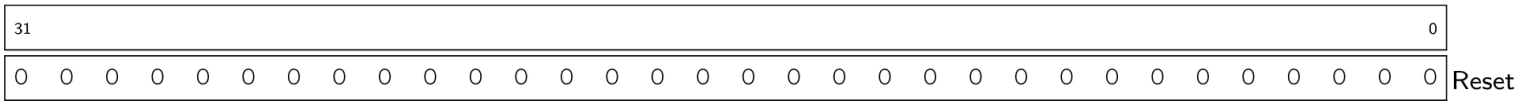
SPI_USR_MOSI_DBITLEN It indicates the length of MOSI data minus one, in multiples of one bit. It is only valid when SPI_USR_MOSI is set to 1 in master mode. (R/W)

Register 20.12. SPI_MISO_DLEN_REG (0x2C)



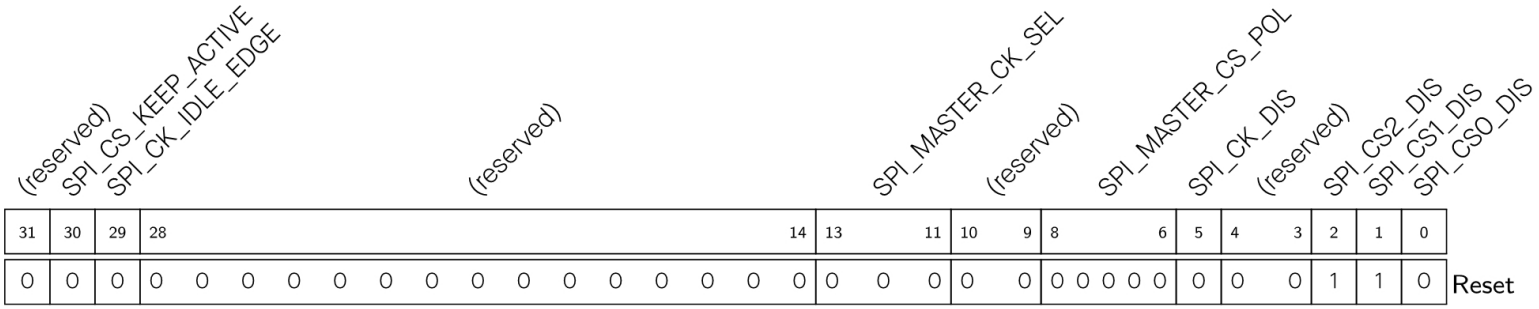
SPI_USR_MISO_DBITLEN It indicates the length of MISO data minus one, in multiples of one bit. It is only valid when SPI_USR_MISO is set to 1 in master mode. (R/W)

Register 20.13. SPI_SLV_WR_STATUS_REG (0x30)



SPI_SLV_WR_STATUS_REG In the slave mode this register is the status register for the master to write the slave. In the master mode, if the address length is bigger than 32 bits, SPI_ADDR_REG stores the higher 32 bits of address value, and this register stores the rest lower part of address value. (R/W)

Register 20.14. SPI_PIN_REG (0x34)



SPI_CS_KEEP_ACTIVE This bit is only used in master mode where when it is set, the CS signal will keep active. (R/W)

SPI_CK_IDLE_EDGE This bit is only used in master mode to configure the logicl level of SPI output clock in idle state. (R/W)
1: the spi_clk line keeps high when idle;
0: the spi_clk line keeps low when idle.

SPI_MASTER_CK_SEL Reserved.

SPI_MASTER_CS_POL Reserved.

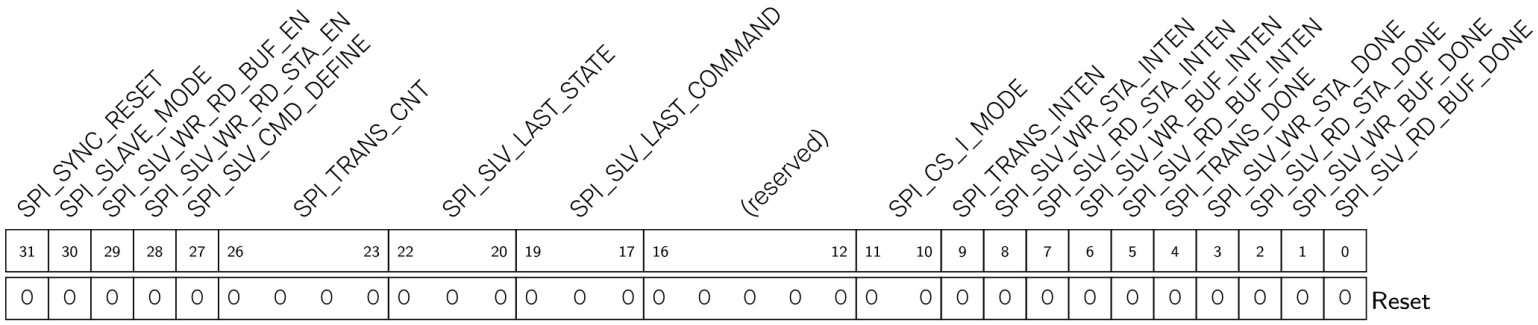
SPI_CK_DIS Reserved.

SPI_CS2_DIS This bit enables the SPI CS2 signal. 1: disables CS2; 0: enables CS2. (R/W)

SPI_CS1_DIS This bit enables the SPI CS1 signal. 1: disables CS1; 0: enables CS1. (R/W)

SPI_CS0_DIS This bit enables the SPI CS0 signal. 1: disables CS0; 0: enables CS0. (R/W)

Register 20.15. SPI_SLAVE_REG (0x38)



SPI_SYNC_RESET When set, it resets the latched values of the SPI clock line, CS line and data line. (R/W)

SPI_SLAVE_MODE This bit is used to set the mode of the SPI device. (R/W)

1: slave mode;

0: master mode.

SPI_SLV_WR_RD_BUF_EN This bit is only used in slave half-duplex mode, where when it is set, the write and read data commands are enabled. (R/W)

SPI_SLV_WR_RD_STA_EN This bit is only used in slave half-duplex mode, where when it is set, the write and read status commands are enabled. (R/W)

SPI_SLV_CMD_DEFINE Reserved.

SPI_TRANS_CNT The counter for operations in both the master mode and the slave mode. (RO)

SPI_SLV_LAST_STATE In slave mode, this contains the state of the SPI state machine. (RO)

SPI_SLV_LAST_COMMAND Reserved.

SPI_CS_I_MODE Reserved.

SPI_TRANS_INTEN The interrupt enable bit for the SPI_TRANS_DONE_INT interrupt. (R/W)

SPI_SLV_WR_STA_INTEN The interrupt enable bit for the SPI_SLV_WR_STA_INT interrupt. (R/W)

SPI_SLV_RD_STA_INTEN The interrupt enable bit for the SPI_SLV_RD_STA_INT interrupt. (R/W)

SPI_SLV_WR_BUF_INTEN The interrupt enable bit for the SPI_SLV_WR_BUF_INT interrupt. (R/W)

SPI_SLV_RD_BUF_INTEN The interrupt enable bit for the SPI_SLV_RD_BUF_INT interrupt. (R/W)

SPI_TRANS_DONE The raw interrupt status bit for the SPI_TRANS_DONE_INT interrupt. It is set by hardware and cleared by software. (R/W)

SPI_SLV_WR_STA_DONE The raw interrupt status bit for the SPI_SLV_WR_STA_INT interrupt. It is set by hardware and cleared by software, and only applicable to slave half-duplex mode. (R/W)

SPI_SLV_RD_STA_DONE The raw interrupt status bit for the SPI_SLV_RD_STA_INT interrupt. It is set by hardware and cleared by software, and only applicable to slave half-duplex mode. (R/W)

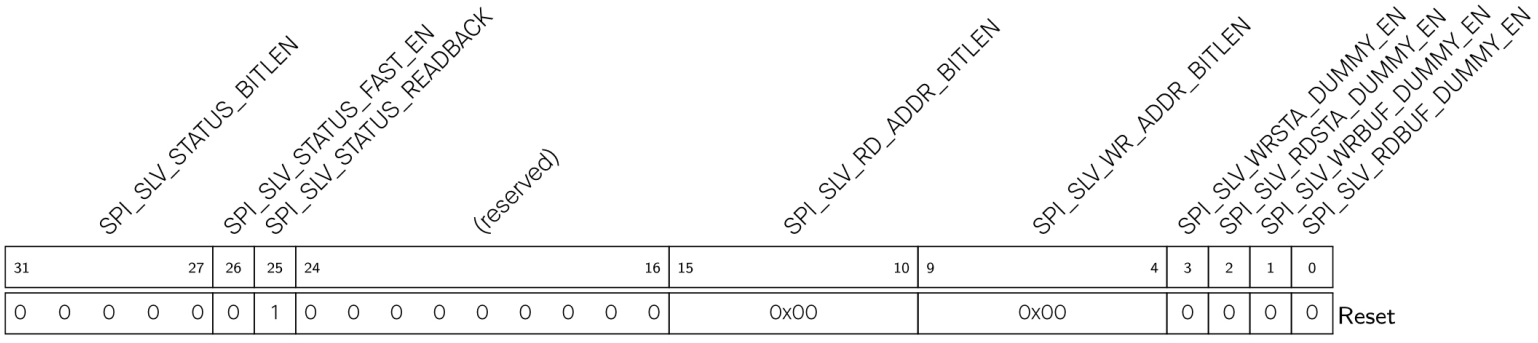
Register 20.15. SPI_SLAVE_REG (0x38)

Continued from the previous page ...

SPI_SLV_WR_BUF_DONE The raw interrupt status bit for the SPI_SLV_WR_BUF_INT interrupt. It is set by hardware and cleared by software, and only applicable to slave half-duplex mode. (R/W)

SPI_SLV_RD_BUF_DONE The raw interrupt status bit for the SPI_SLV_RD_BUF_INT interrupt. It is set by hardware and cleared by software, and only applicable to slave half-duplex mode. (R/W)

Register 20.16. SPI_SLAVE1_REG (0x3C)



SPI_SLV_STATUS_BITLEN It is only used in slave half-duplex mode to configure the length of the master writing into the status register. (R/W)

SPI_SLV_STATUS_FAST_EN Reserved.

SPI_SLV_STATUS_READBACK Reserved.

SPI_SLV_RD_ADDR_BITLEN It indicates the address length in bits minus one for a slave-read operation. It is only valid in slave half-duplex mode. (R/W)

SPI_SLV_WR_ADDR_BITLEN It indicates the address length in bits minus one for a slave-write operation. It is only valid in slave half-duplex mode. (R/W)

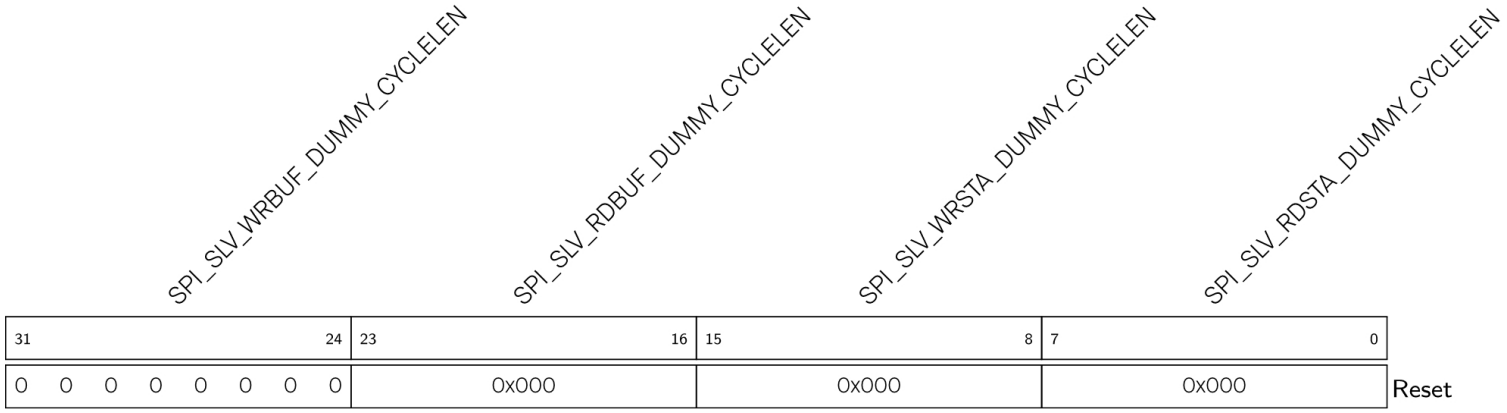
SPI_SLV_WRSTA_DUMMY_EN In slave mode, this bit enables the dummy phase for write-status operations. It is only valid in slave half-duplex mode.(R/W)

SPI_SLV_RDSTA_DUMMY_EN In slave mode, this bit enables the dummy phase for read-status operations. It is only valid in slave half-duplex mode. (R/W)

SPI_SLV_WRBUF_DUMMY_EN In slave mode, this bit enables the dummy phase for write-buffer operations. It is only valid in slave half-duplex mode. (R/W)

SPI_SLV_RDBUF_DUMMY_EN In slave mode, this bit enables the dummy phase for read-buffer operations. It is only valid in slave half-duplex mode. (R/W)

Register 20.17. SPI_SLAVE2_REG (0x40)



SPI_SLV_WRBUF_DUMMY_CYCLELEN It indicates the number of SPI clock cycles minus one for the dummy phase for write-data operations. It is only valid when SPI_SLV_WRBUF_DUMMY_EN is set to 1 in slave half-duplex mode. (R/W)

SPI_SLV_RDBUF_DUMMY_CYCLELEN It indicates the number of SPI clock cycles minus one for the dummy phase for read-data operations. It is only valid when SPI_SLV_RDBUF_DUMMY_EN is set to 1 in slave half-duplex mode. (R/W)

SPI_SLV_WRSTA_DUMMY_CYCLELEN It indicates the number of SPI clock cycles minus one for the dummy phase for write-status register operations. It is only valid when SPI_SLV_WRSTA_DUMMY_EN is set to 1 in slave half-duplex mode. (R/W)

SPI_SLV_RDSTA_DUMMY_CYCLELEN It indicates the number of SPI clock cycles minus one for the dummy phase for read-status register operations. It is only valid when SPI_SLV_RDSTA_DUMMY_EN is set to 1 in slave half-duplex mode. (R/W)

Register 20.18. SPI_SLAVE3_REG (0x44)

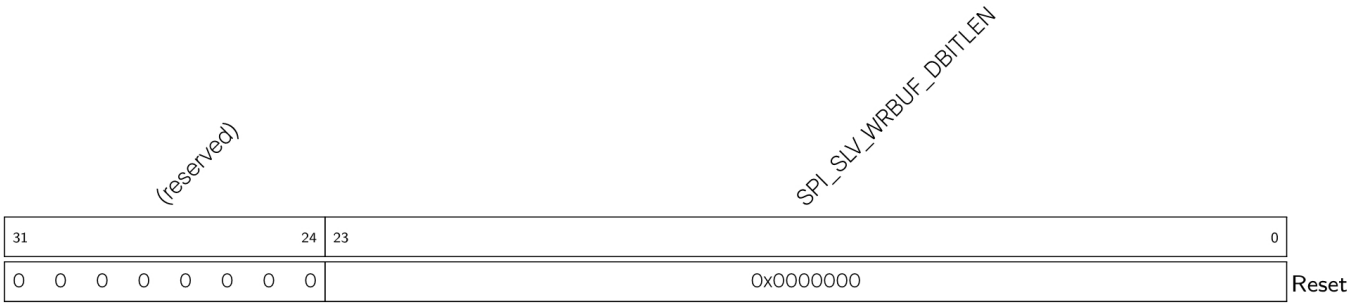
SPI_SLV_WRSTA_CMD_VALUE Reserved.

SPI_SLV_RDSTA_CMD_VALUE Reserved.

SPI_SLV_WRBUF_CMD_VALUE Reserved.

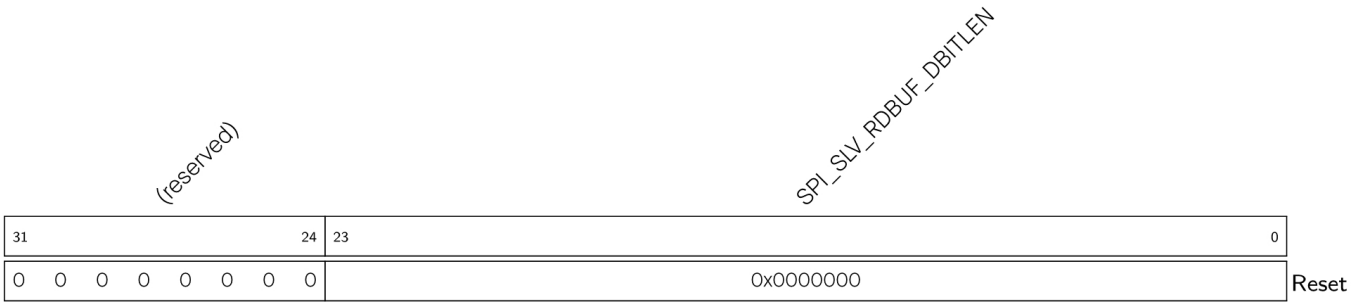
SPI_SLV_RDBUF_CMD_VALUE Reserved.

Register 20.19. SPI_SLV_WRBUF_DLEN_REG (0x48)



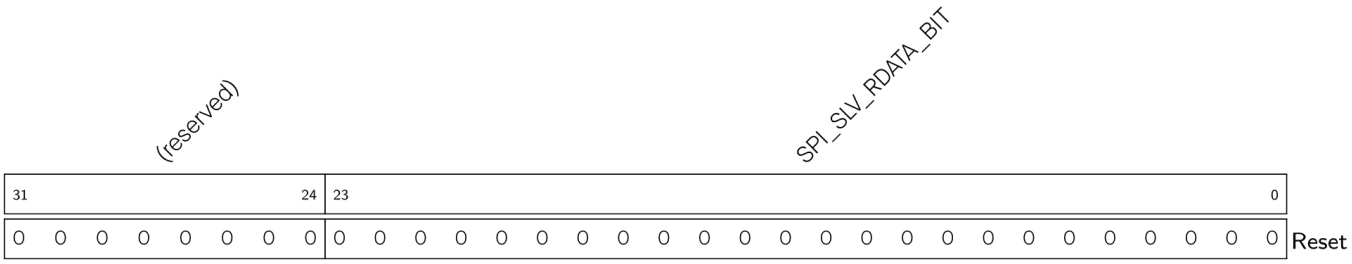
SPI_SLV_WRBUF_DBITLEN It indicates the length of written data minus one, in multiples of one bit. It is only valid in slave half-duplex mode. (R/W)

Register 20.20. SPI_SLV_RDBUF_DLEN_REG (0x4C)



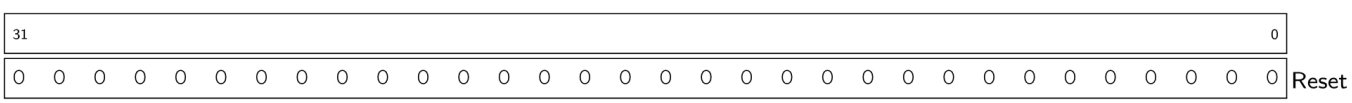
SPI_SLV_RDBUF_DBITLEN It indicates the length of read data minus one, in multiples of one bit. It is only valid in slave half-duplex mode. (R/W)

Register 20.21. SPI_SLV_RD_BIT_REG (0x64)



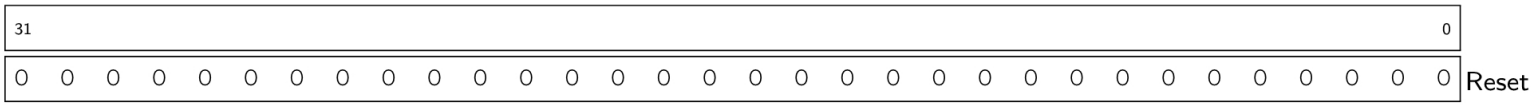
SPI_SLV_RDATA_BIT It indicates the bit length of data the master reads from the slave, minus one. It is only valid in slave half-duplex mode. (R/W)

Register 20.22. SPI_W*n*_REG (*n*: 0-15) (0x80+4**n*)



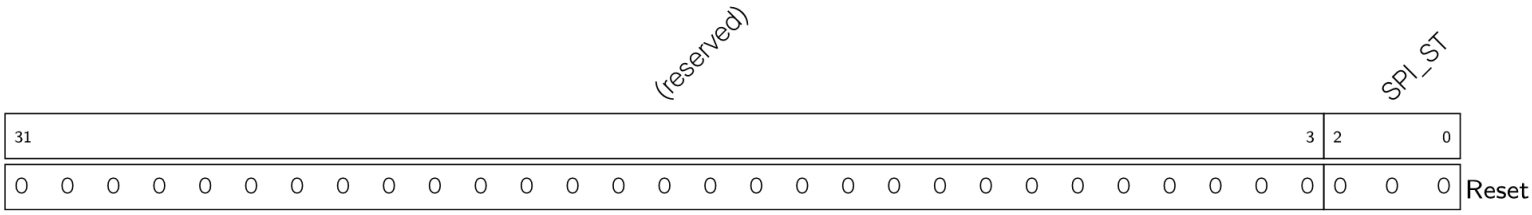
SPI_W*n*_REG Data buffer. (R/W)

Register 20.23. SPI_TX_CRC_REG (0xC0)



SPI_TX_CRC_REG Reserved.

Register 20.24. SPI_EXT2_REG (0xF8)



SPI_ST The current state of the SPI state machine: (RO)

0: idle state

1: preparation state

2: send command state

3: send data state

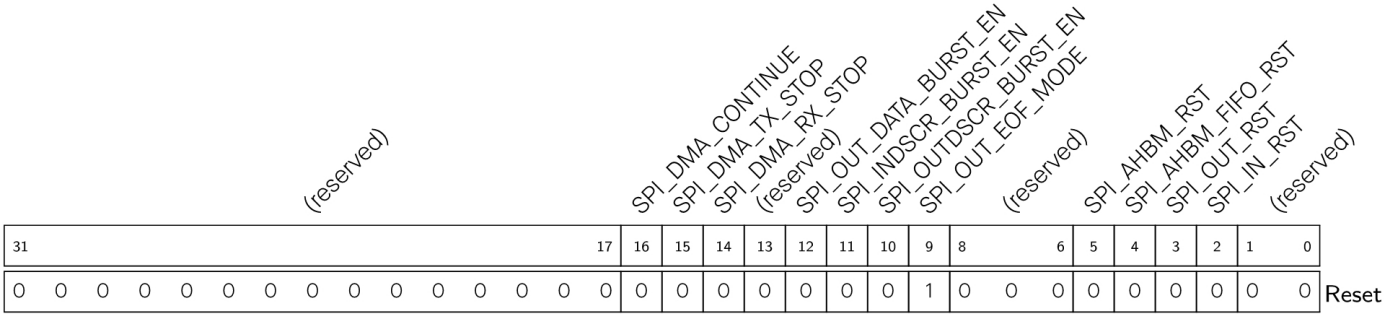
4: read data state

5: write data state

6: wait state

7: done state

Register 20.25. SPI_DMA_CONF_REG (0x100)



SPI_DMA_CONTINUE This bit enables SPI DMA continuous data TX/RX mode. (R/W)

SPI_DMA_TX_STOP When in continuous TX/RX mode, setting this bit stops sending data. (R/W)

SPI_DMA_RX_STOP When in continuous TX/RX mode, setting this bit stops receiving data. (R/W)

SPI_OUT_DATA_BURST_EN SPI DMA reads data from memory in burst mode. (R/W)

SPI_INDSCR_BURST_EN SPI DMA reads inlink descriptor in burst mode. (R/W)

SPI_OUTDSCR_BURST_EN SPI DMA reads outlink descriptor in burst mode. (R/W)

SPI_OUT_EOF_MODE DMA out-EOF-flag generation mode. (R/W)

1: out-EOF-flag is generated when DMA has popped all data from the FIFO;

0: out-EOF-flag is generated when DMA has pushed all data to the FIFO.

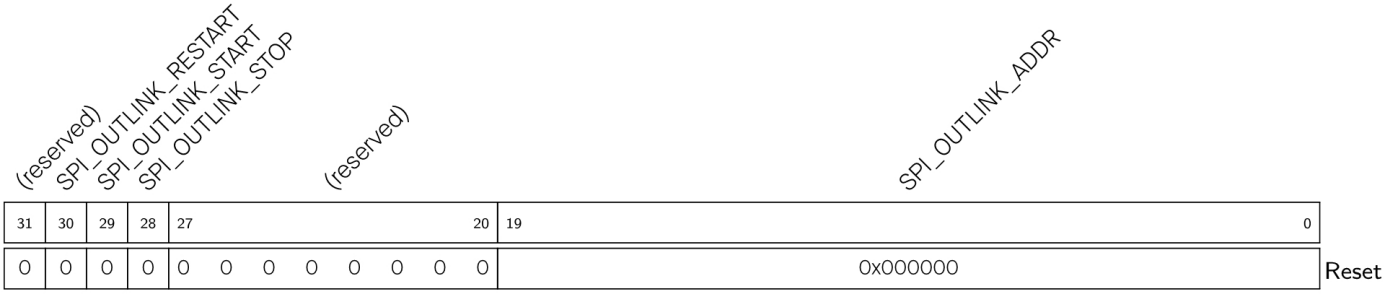
SPI_AHBM_RST reset SPI DMA AHB master. (R/W)

SPI_AHBM_FIFO_RST This bit is used to reset SPI DMA AHB master FIFO pointer. (R/W)

SPI_OUT_RST The bit is used to reset DMA out-FSM and out-data FIFO pointer. (R/W)

SPI_IN_RST The bit is used to reset DMA in-DSM and in-data FIFO pointer. (R/W)

Register 20.26. SPI_DMA_OUT_LINK_REG (0x104)



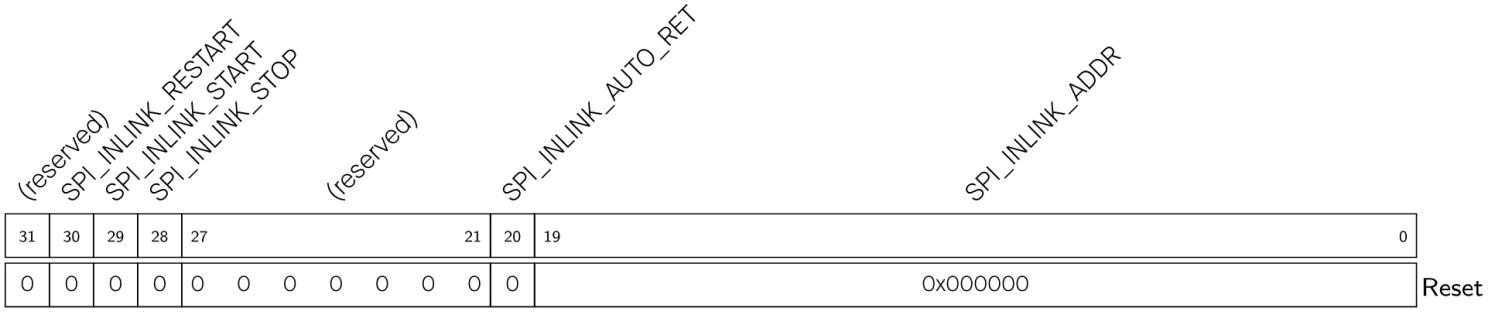
SPI_OUTLINK_RESTART Set the bit to add new outlink descriptors. (R/W)

SPI_OUTLINK_START Set the bit to start to use outlink descriptor. (R/W)

SPI_OUTLINK_STOP Set the bit to stop to use outlink descriptor. (R/W)

SPI_OUTLINK_ADDR The address of the first outlink descriptor. (R/W)

Register 20.27. SPI_DMA_IN_LINK_REG (0x108)



SPI_INLINK_RESTART Set the bit to add new inlink descriptors. (R/W)

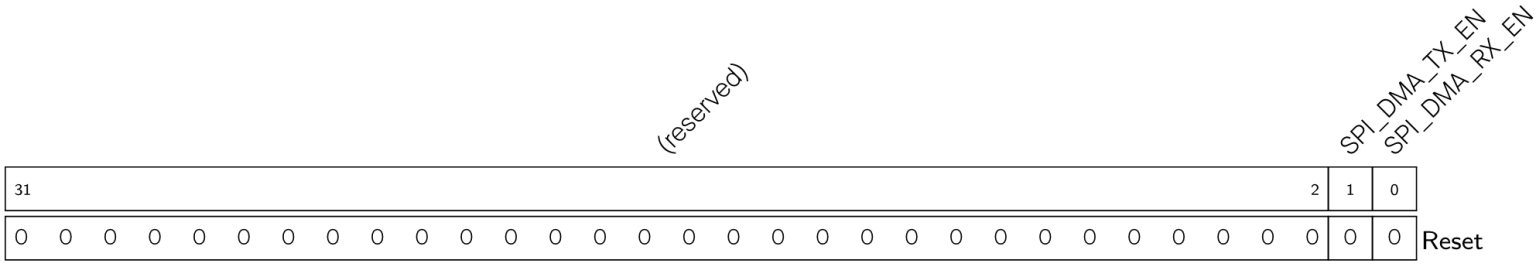
SPI_INLINK_START Set the bit to start to use inlink descriptor. (R/W)

SPI_INLINK_STOP Set the bit to stop to use inlink descriptor. (R/W)

SPI_INLINK_AUTO_RET when the bit is set, inlink descriptor jumps to the next descriptor when a packet is invalid. (R/W)

SPI_INLINK_ADDR The address of the first inlink descriptor. (R/W)

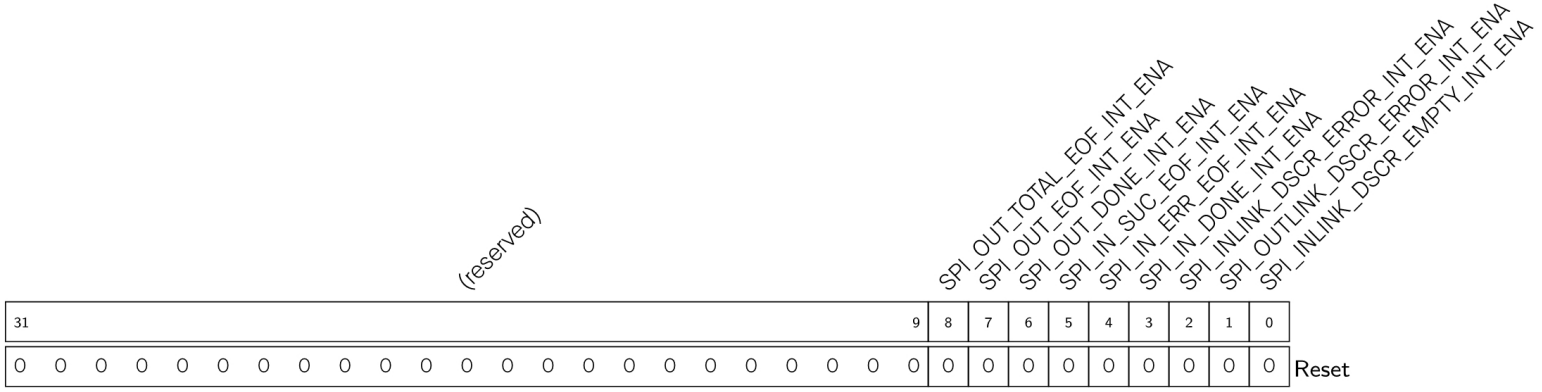
Register 20.28. SPI_DMA_STATUS_REG (0x10C)



SPI_DMA_TX_EN SPI DMA write-data status bit. (RO)

SPI_DMA_RX_EN SPI DMA read-data status bit. (RO)

Register 20.29. SPI_DMA_INT_ENA_REG (0x110)



SPI_OUT_TOTAL_EOF_INT_ENA The interrupt enable bit for the SPI_OUT_TOTAL_EOF_INT interrupt. (R/W)

SPI_OUT_EOF_INT_ENA The interrupt enable bit for the SPI_OUT_EOF_INT interrupt. (R/W)

SPI_OUT_DONE_INT_ENA The interrupt enable bit for the SPI_OUT_DONE_INT interrupt. (R/W)

SPI_IN_SUC_EOF_INT_ENA The interrupt enable bit for the SPI_IN_SUC_EOF_INT interrupt. (R/W)

SPI_IN_ERR_EOF_INT_ENA The interrupt enable bit for the SPI_IN_ERR_EOF_INT interrupt. (R/W)

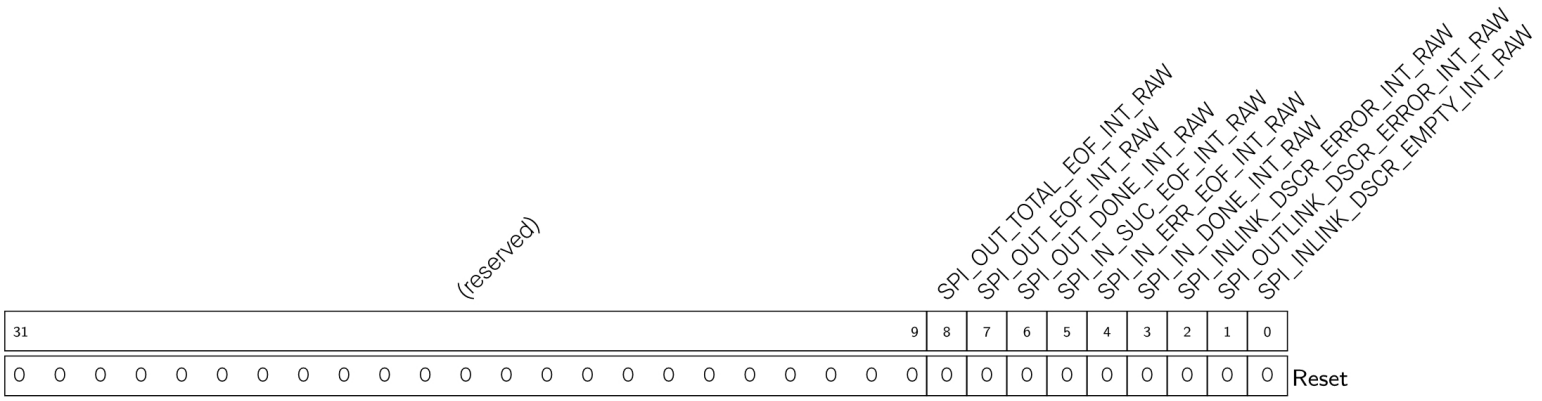
SPI_IN_DONE_INT_ENA The interrupt enable bit for the SPI_IN_DONE_INT interrupt. (R/W)

SPI_INLINK_DSCR_ERROR_INT_ENA The interrupt enable bit for the SPI_INLINK_DSCR_ERROR_INT interrupt. (R/W)

SPI_OUTLINK_DSCR_ERROR_INT_ENA The interrupt enable bit for the SPI_OUTLINK_DSCR_ERROR_INT interrupt. (R/W)

SPI_INLINK_DSCR_EMPTY_INT_ENA The interrupt enable bit for the SPI_INLINK_DSCR_EMPTY_INT interrupt. (R/W)

Register 20.30. SPI_DMA_INT_RAW_REG (0x114)



SPI_OUT_TOTAL_EOF_INT_RAW The raw interrupt status bit for the SPI_OUT_TOTAL_EOF_INT interrupt. (RO)

SPI_OUT_EOF_INT_RAW The raw interrupt status bit for the SPI_OUT_EOF_INT interrupt. (RO)

SPI_OUT_DONE_INT_RAW The raw interrupt status bit for the SPI_OUT_DONE_INT interrupt. (RO)

SPI_IN_SUC_EOF_INT_RAW The raw interrupt status bit for the SPI_IN_SUC_EOF_INT interrupt. (RO)

SPI_IN_ERR_EOF_INT_RAW The raw interrupt status bit for the SPI_IN_ERR_EOF_INT interrupt. (RO)

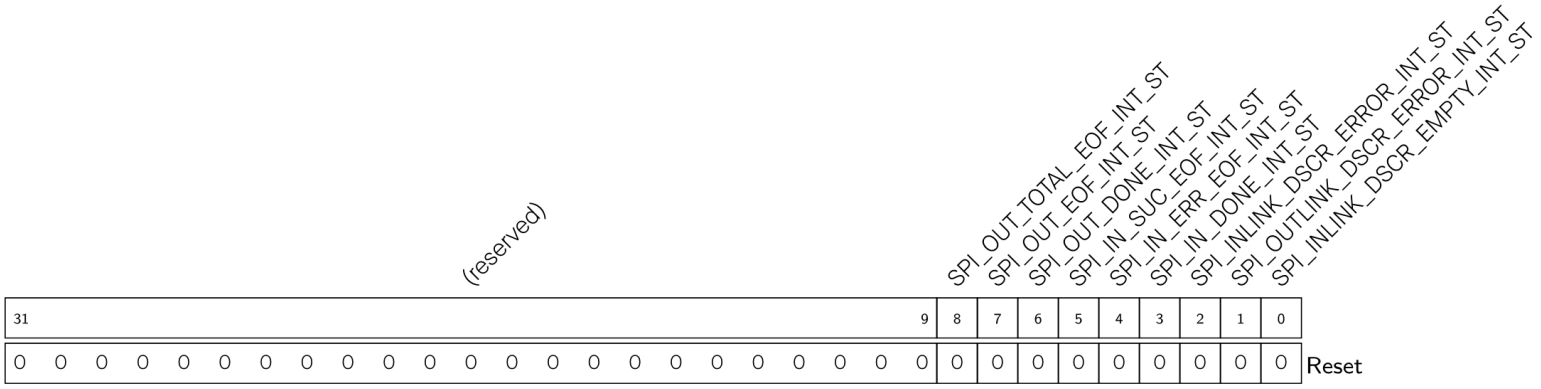
SPI_IN_DONE_INT_RAW The raw interrupt status bit for the SPI_IN_DONE_INT interrupt. (RO)

SPI_INLINK_DSCR_ERROR_INT_RAW The raw interrupt status bit for the SPI_INLINK_DSCR_ERROR_INT interrupt. (RO)

SPI_OUTLINK_DSCR_ERROR_INT_RAW The raw interrupt status bit for the SPI_OUTLINK_DSCR_ERROR_INT interrupt. (RO)

SPI_INLINK_DSCR_EMPTY_INT_RAW The raw interrupt status bit for the SPI_INLINK_DSCR_EMPTY_INT interrupt. (RO)

Register 20.31. SPI_DMA_INT_ST_REG (0x118)



SPI_OUT_TOTAL_EOF_INT_ST The masked interrupt status bit for the SPI_OUT_TOTAL_EOF_INT interrupt. (RO)

SPI_OUT_EOF_INT_ST The masked interrupt status bit for the SPI_OUT_EOF_INT interrupt. (RO)

SPI_OUT_DONE_INT_ST The masked interrupt status bit for the SPI_OUT_DONE_INT interrupt. (RO)

SPI_IN_SUC_EOF_INT_ST The masked interrupt status bit for the SPI_IN_SUC_EOF_INT interrupt. (RO)

SPI_IN_ERR_EOF_INT_ST The masked interrupt status bit for the SPI_IN_ERR_EOF_INT interrupt. (RO)

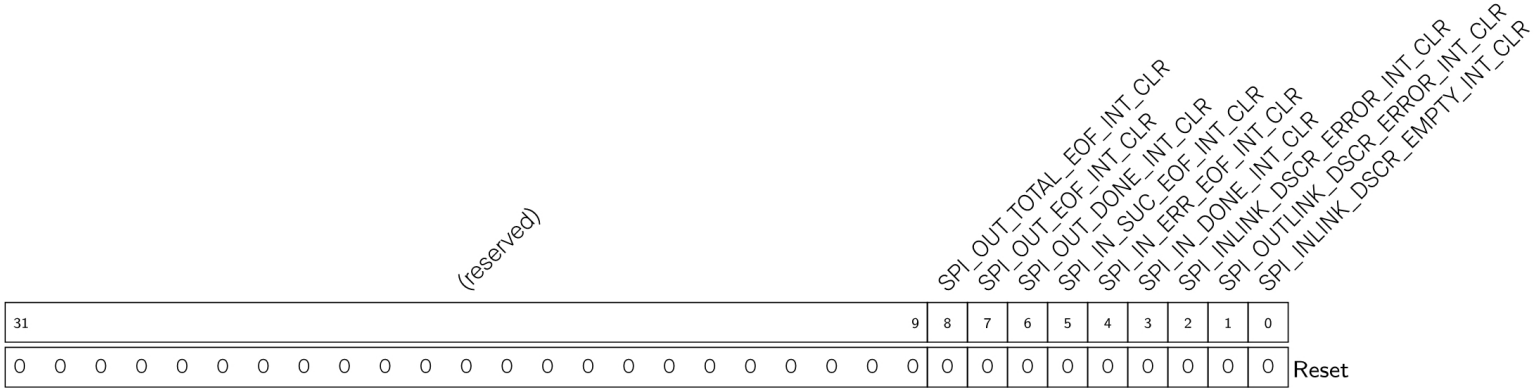
SPI_IN_DONE_INT_ST The masked interrupt status bit for the SPI_IN_DONE_INT interrupt. (RO)

SPI_INLINK_DSCR_ERROR_INT_ST The masked interrupt status bit for the SPI_INLINK_DSCR_ERROR_INT interrupt. (RO)

SPI_OUTLINK_DSCR_ERROR_INT_ST The masked interrupt status bit for the SPI_OUTLINK_DSCR_ERROR_INT interrupt. (RO)

SPI_INLINK_DSCR_EMPTY_INT_ST The masked interrupt status bit for the SPI_INLINK_DSCR_EMPTY_INT interrupt. (RO)

Register 20.32. SPI_DMA_INT_CLR_REG (0x11C)



SPI_OUT_TOTAL_EOF_INT_CLR Set this bit to clear the SPI_OUT_TOTAL_EOF_INT interrupt. (R/W)

SPI_OUT_EOF_INT_CLR Set this bit to clear the SPI_OUT_EOF_INT interrupt. (R/W)

SPI_OUT_DONE_INT_CLR Set this bit to clear the SPI_OUT_DONE_INT interrupt. (R/W)

SPI_IN_SUC_EOF_INT_CLR Set this bit to clear the SPI_IN_SUC_EOF_INT interrupt. (R/W)

SPI_IN_ERR_EOF_INT_CLR Set this bit to clear the SPI_IN_ERR_EOF_INT interrupt. (R/W)

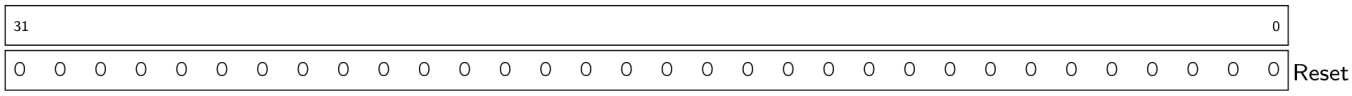
SPI_IN_DONE_INT_CLR Set this bit to clear the SPI_IN_DONE_INT interrupt. (R/W)

SPI_INLINK_DSCR_ERROR_INT_CLR Set this bit to clear the SPI_INLINK_DSCR_ERROR_INT interrupt. (R/W)

SPI_OUTLINK_DSCR_ERROR_INT_CLR Set this bit to clear the SPI_OUTLINK_DSCR_ERROR_INT interrupt. (R/W)

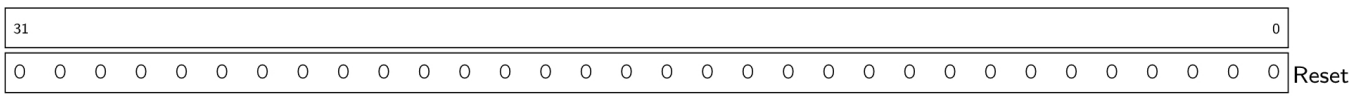
SPI_INLINK_DSCR_EMPTY_INT_CLR Set this bit to clear the SPI_INLINK_DSCR_EMPTY_INT interrupt. (R/W)

Register 20.33. SPI_IN_ERR_EOF_DES_ADDR_REG (0x120)



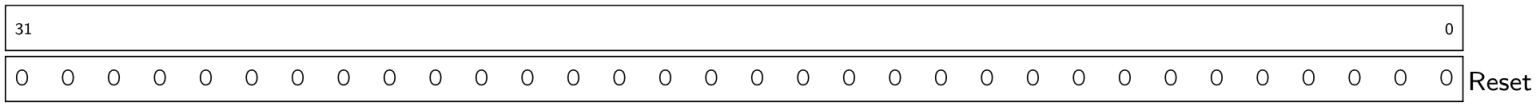
SPI_IN_ERR_EOF_DES_ADDR_REG The inlink descriptor address when SPI DMA encountered an error in receiving data. (RO)

Register 20.34. SPI_IN_SUC_EOF_DES_ADDR_REG (0x124)



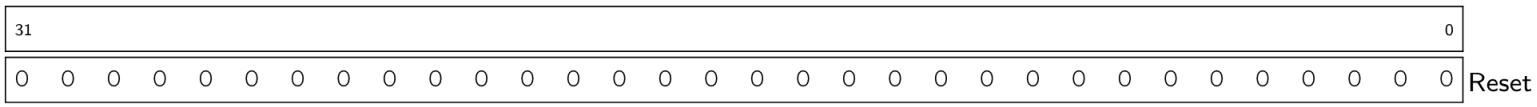
SPI_IN_SUC_EOF_DES_ADDR_REG The last inlink descriptor address when SPI DMA encountered EOF. (RO)

Register 20.35. SPI_INLINK_DSCR_REG (0x128)



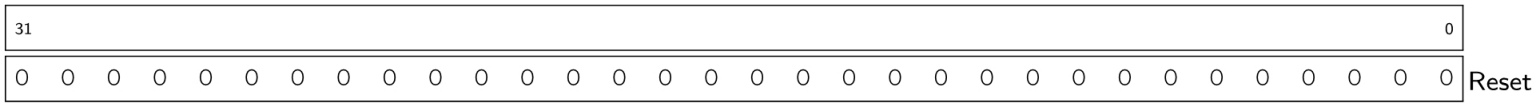
SPI_INLINK_DSCR_REG The address of the current inlink descriptor. (RO)

Register 20.36. SPI_INLINK_DSCR_BF0_REG (0x12C)



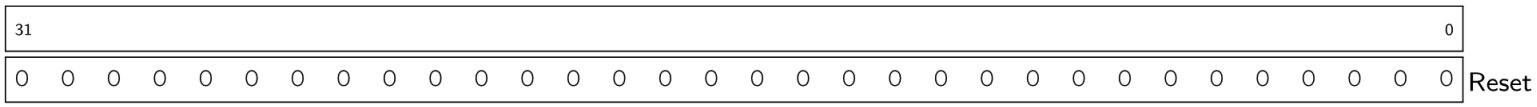
SPI_INLINK_DSCR_BF0_REG The address of the next inlink descriptor. (RO)

Register 20.37. SPI_INLINK_DSCR_BF1_REG (0x130)



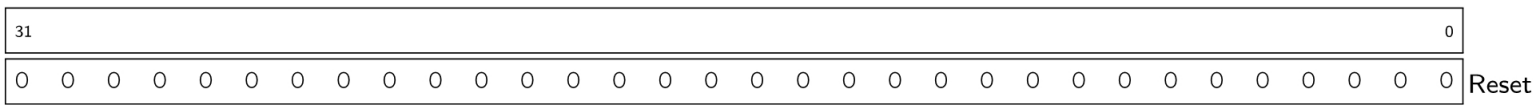
SPI_INLINK_DSCR_BF1_REG The address of the next inlink data buffer. (RO)

Register 20.38. SPI_OUT_EOF_BFR_DES_ADDR_REG (0x134)



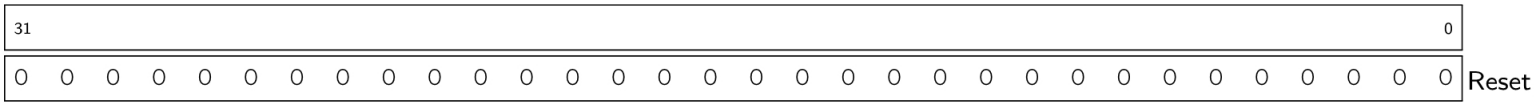
SPI_OUT_EOF_BFR_DES_ADDR_REG The buffer address corresponding to the outlink descriptor that produces EOF. (RO)

Register 20.39. SPI_OUT_EOF_DES_ADDR_REG (0x138)



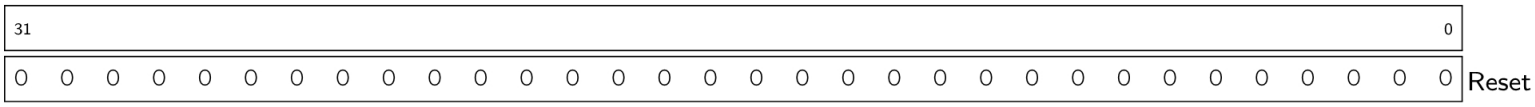
SPI_OUT_EOF_DES_ADDR_REG The last outlink descriptor address when SPI DMA encountered EOF. (RO)

Register 20.40. SPI_OUTLINK_DSCR_REG (0x13C)



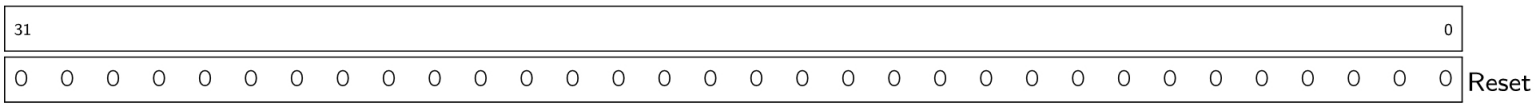
SPI_OUTLINK_DSCR_REG The address of the current outlink descriptor. (RO)

Register 20.41. SPI_OUTLINK_DSCR_BF0_REG (0x140)



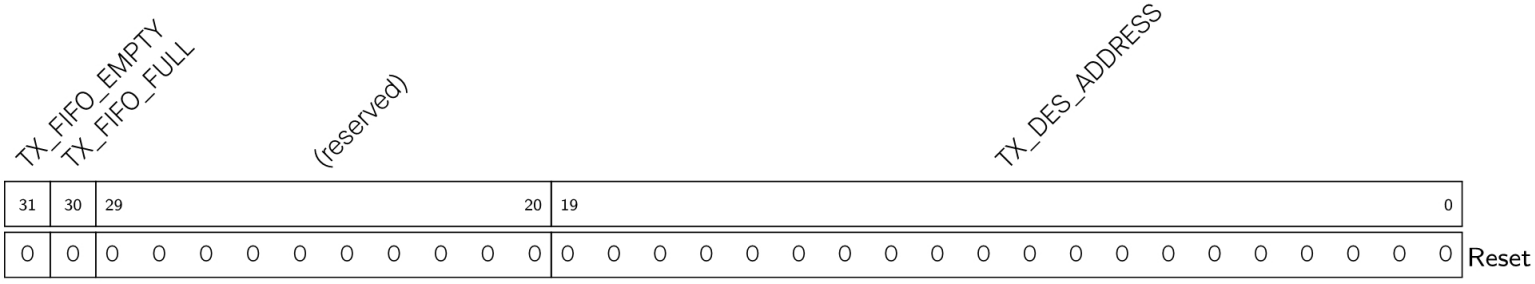
SPI_OUTLINK_DSCR_BF0_REG The address of the next outlink descriptor. (RO)

Register 20.42. SPI_OUTLINK_DSCR_BF1_REG (0x144)



SPI_OUTLINK_DSCR_BF1_REG The address of the next outlink data buffer. (RO)

Register 20.43. SPI_DMA_RSTATUS_REG (0x148)

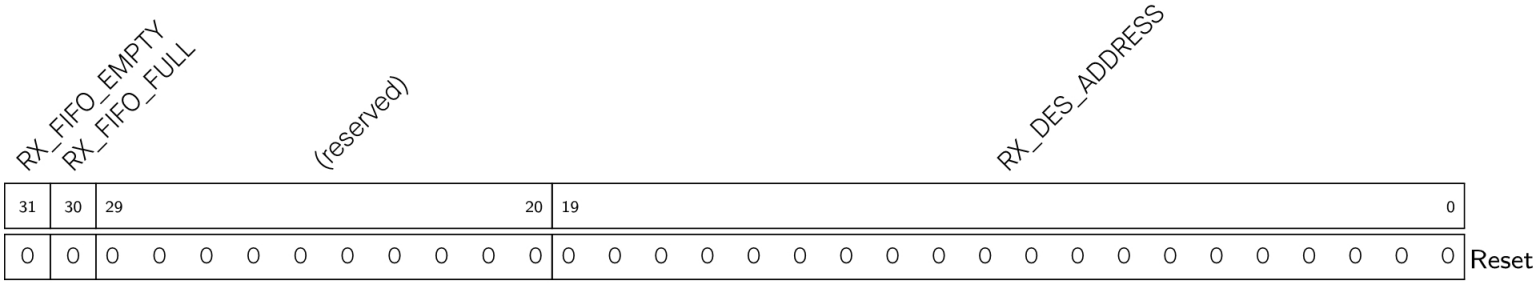


TX_FIFO_EMPTY The SPI DMA TX FIFO is empty. (RO)

TX_FIFO_FULL The SPI DMA TX FIFO is full. (RO)

TX_DES_ADDRESS The LSB of the SPI DMA outlink descriptor address. (RO)

Register 20.44. SPI_DMA_TSTATUS_REG (0x14C)



RX_FIFO_EMPTY The SPI DMA RX FIFO is empty. (RO)

RX_FIFO_FULL The SPI DMA RX FIFO is full. (RO)

RX_DES_ADDRESS The LSB of the SPI DMA inlink descriptor address. (RO)