1. Summary

Acting as a peripheral, the SPI module provides fast serial communication capabilities to the Airi5c processor. After a complete redesign, this Module now supports the following features:

- AHB-Lite interface
- Separate registers for control, rx and tx status, all with set/clear access capability
- configurable rx/tx FIFO size (1 - 256 frames)
- configurable number of data bits
- master and slave support
- 4 slave select pins
- Full asynchronous Slave design
- accessible rx and tx FIFO fill levels
- configurable and independent watermark settings for rx and tx FIFO fill level with interrupt generation
- error detection
- extensive interrupt capabilities

2. Parameters

These parameters have to be set at synthesis, they cannot be changed at runtime.

Parameter	Default	Description
BASE_ADDR	0xC0000500	Base address of the SPI module, the addresses of all registers are increments of 4 beginning at this address
MASTER_ON_RESET	0	Defines whether the module acts as a master or slave after reset
ADDR_WIDTH	2	Address width of the tx/rx FIFO, defining the max fill level ($size = 2^{width}$)
DATA_WIDTH	8	Defines the word the word length and accordingly the number of bits in the internal tx/rx shift registers

3. Registers

The SPI module includes the following 10 32-bit data, control and status registers, which can be accessed via AHB-Lite interface.

In the current processor design, there are two SPI modules: SPI0 and SPI1. SPI0 is located at base address 0xC0000400 and is hardwired to QSPI in some FPGA boards including NexysVideo. Make sure to use SPI1 at base address 0xC0000500 instead if other purposes are intended. On reset, SPI0 is configured as a master and SPI1 as a slave device.

Address	Type	Description
BASE_ADDR + 0x00 0xC0000400 (SPI0) 0xC0000500 (SPI1)	DATA	Write access writes to tx FIFO, read access reads from rx FIFO
BASE_ADDR + 0x04 0xC0000404 (SPI0) 0xC0000504 (SPI1)	Ctrl reg	This register contains all communication settings, such as clock divider, polarity phase, slave select and master/slave configuration
BASE_ADDR + 0x08 0xC0000408 (SPI0) 0xC0000508 (SPI1)	Ctrl reg set	Writing to this register automatically sets the specified bits in ctrl reg
BASE_ADDR + 0x0C 0xC000040C (SPI0) 0xC000050C (SPI1)	Ctrl reg clr	Writing to this register automatically clears the specified bits in ctrl reg
BASE_ADDR + 0x10 0xC0000410 (SPI0) 0xC0000510 (SPI1)	Tx stat reg	This register contains the tx status, such as tx FIFO fill level, errors and interrupt enables
BASE_ADDR + 0x14 0xC0000414 (SPI0) 0xC0000514 (SPI1)	Tx stat reg set	Writing to this register automatically sets the specified bits in tx stat reg
BASE_ADDR + 0x18 0xC0000418 (SPI0) 0xC0000518 (SPI1)	Tx stat reg clr	Writing to this register automatically clears the specified bits in tx stat reg
BASE_ADDR + 0x1C 0xC000041C (SPI0) 0xC000051C (SPI1)	Rx stat reg	This register contains the rx status, such as rx FIFO fill level, errors and interrupt enables
BASE_ADDR + 0x20 0xC0000520 (SPI0) 0xC0000520 (SPI1)	Rx stat reg set	Writing to this register automatically sets the specified bits in rx stat reg
BASE_ADDR + 0x24 0xC0000524 (SPI0) 0xC0000524 (SPI1)	Rx stat reg clr	Writing to this register automatically clears the specified bits in rx stat reg

3.1. Control Register

Bits	Access	Description
31:17	r	reserved
16	rw	Defines whether the device is master (1) or slave (0)
15:14	r	reserved
13	rw	Defines whether slave select is driven by master (0) or manual slave select (1)
12	rw	Manual slave select
11:10	r	reserved
9:8	rw	Active slave select (only used in master mode)
7:6	r	reserved
5	rw	Clock polarity
4	rw	Clock phase
3:0	rw	Clock divider ($clk_{div} = 2^{x+1}$)

3.2. Tx status register

Bits	Access	Description
31:28	r	reserved
27	rw	Tx ready interrupt enable
26	rw	Tx overflow error interrupt enable
25	rw	Tx watermark reached interrupt enable
24	rw	Tx FIFO empty interrupt enable
23:21	r	reserved
20	r	Tx ready
19	rw	Tx overflow error (write access when FIFO is full)
18	r	Tx FIFO fill level ≤ tx watermark
17	r	Tx FIFO empty
16	r	Tx FIFO full
15:8	rw	Tx FIFO watermark
7:0	r	Tx FIFO fill level

3.3. Rx status register

Bits	Access	Description
31	rw	Rx ignore
30:28	r	reserved
27	rw	Rx underflow error interrupt enable
26	rw	Rx overflow error interrupt enable
25	rw	Rx watermark reached interrupt enable
24	rw	Rx FIFO full interrupt enable
23:21	r	reserved
20	rw	Rx underflow error (read from empty rx FIFO)
19	rw	Rx overflow error (received data while rx FIFO was full)
18	r	Rx FIFO fill level ≥ rx watermark
17	r	Rx FIFO empty

16	r	Rx FIFO full
15:8	rw	Rx FIFO watermark
7:0	r	Rx FIFO fill level

Reserved fields are hardwired to zero, writing to those fields has no effect. Once set, all errors stay set as long as they get reset manually.

4. Interrupts

The SPI module supports several interrupts, which are stated in the tx and rx status registers. All interrupts are disabled by default and have to be enabled manually if desired. Besides the individual interrupt signals, there is also a special signal “int_any” available at the port of this module which is set whenever at least one interrupt has occurred. Some interrupt signals are connected to the specific error signals. In this case an interrupt service routine has to reset the specific error flag, otherwise the interrupt will fire again and again.

5. Functionality

Transmitting data can be achieved writing to the DATA address, which effectively writes to the tx FIFO. As long as the tx FIFO is not full, new data can be written to it immediately in a row. The SPI module then reads the data in the tx FIFO automatically and transmits it via the mosi pin in master or the miso pin in slave mode. Transactions are initiated by the master. Data written to the tx FIFO of the slave device is hold, until data from the master is received, meaning, for each frame sent, one frame is received. If the tx FIFO of the slave is empty, zeros are transmitted instead. Incoming data is automatically written to the rx FIFO, which can be read from by reading from the DATA address. To ignore any incoming data, the rx ignore flag can be set. As long as the rx FIFO is not full, data can be received. As soon as the rx FIFO is full, any incoming data is lost and the rx overflow error is set. Due to the asynchronous slave design, data transmission is always triggered on clock edges of the master clock, allowing high data rates and an idle slave clock.