SPI Slave + MUX

IP Core Data Sheet

Features:

- IHP SG13G2 process
- 6 output registers, 2 input registers (each 8 bit)
- Addressable SPI with 3 address bits
- Burst mode
- MISO multiplexer to share the SPI bus among different SPI slave instances on one IC

Contact:

IHP Solutions GmbH
Im Technologiepark 7
15236 Frankfurt (Oder)
Germany

Overview

The design “HumaticsSlv” is a digital hard macro in IHP 0.13 µm CMOS technology for an addressable SPI slave providing 6 output registers and 2 input ports, each 8 bits wide.

Additionally, the design “HumaticsMux” is a digital hard macro for a SPI MISO multiplexer, allowing the connection of up to 8 SPI slave instances to one MISO output pad.

Version: 1.1

Description of functionality

The SPI macro implements a common SPI slave device with 6 read-write registers and 2 read-only registers. The registers are mapped at the following register addresses:

Addr	Type	Name	Default value
0x00	read-only	Register 0 / soft-reset(*)	-
0x01	read-only	Register 1	-
0x02	read/write	Register 2	0x11
0x03	read/write	Register 3	0x11
0x04	read/write	Register 4	0x11
0x05	read/write	Register 5	0x11
0x06	read/write	Register 6	0x11
0x07	read/write	Register 7	0x11

(*) Writing any value to register 0x00 will reset register 0x02 to 0x07 to their default value.

The SPI slave has 3 external address input lines to define the SPI bus address of the current instance. The SPI slave will execute an SPI command (and drive the MISO output), if the 3-bit external address in the SPI word equals the setting of the address input port “addr” of the SPI block. This input port needs to be connected to external pads or internally tied high or low to specify the address of the current instance.

The SPI word structure is as follows (MSB first):

1 bit	write_strobe (high-active)
3 bits	external address
1 bit	reserved / future use (set to 0)
3 bits	register address
Nx8 bits	data

It is allowed to write (or read) any number of data bytes / registers in one SPI command (= SPI burst mode). The register address will be automatically incremented (chip address does not change). The SPI transfer ends when the chip select line CS is de-asserted.

The SPI operates in SPI mode CPOL=0 and CPHA=0, i.e. the clock (SCLK) is low in idle state and only toggling during SPI data transfers. MOSI and MISO change on the falling SCLK edge. The waveform is shown in the following diagram. Please note that the drawn MISO waveform is the final waveform on the SPI bus if the MISO and MISO_OE pin of the SPI block are properly connected to the MISO output pad.

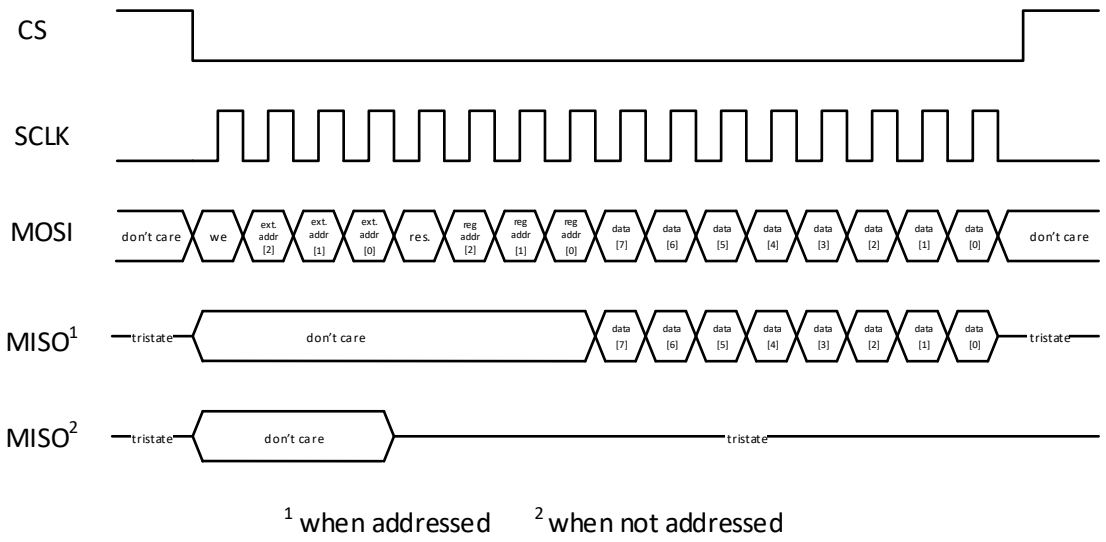

Figure 1: SPI Waveform

Upon write access, MISO will return the old register content, not reproduce the new one.

Upon read access, the data on MOSI after the register address are “don’t care”.

The additional SPI multiplexer allows the connection of up to 8 SPI slave instances to one MISO output pad. If less than 8 SPI instances are connected to the multiplexer, unused input ports miso_in and oen_in of the multiplexer must be connected to logic high (1.2 V) to ensure functionality.

Specifications

Supply voltages

Symbol	Parameter	Min	Typ	Max	Unit
V_{ddcore}	Digital core supply		1.2		V

Logic levels

Symbol	Parameter	Min	Typ	Max	Unit
V_{high}	High-level input / output voltage		1.2		V
V_{low}	Low-level input / output voltage		0		V

Timing performance

Symbol	Parameter	Typ	Unit
f_{sclk}	Max. operating SCLK frequency of SPI core without IO pads, signal lines etc.	50	MHz

Environmental conditions

Symbol	Parameter	Min	Max	Unit
T_{op}	Operating temperature	0	40	°C

Pin arrangement / Dimensions

The SPI slave occupies a silicon area of 189.94 µm x 65.1 µm. The SPI MUX occupies 10.08 µm x 39.9 µm.

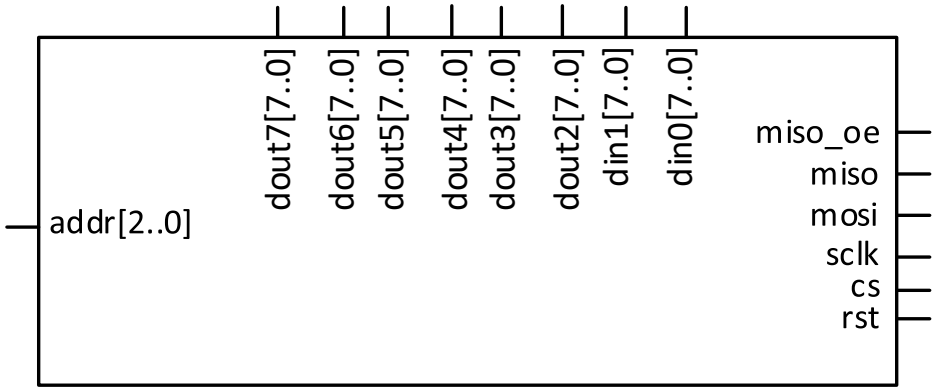

Figure 2: Pin arrangement SPI slave

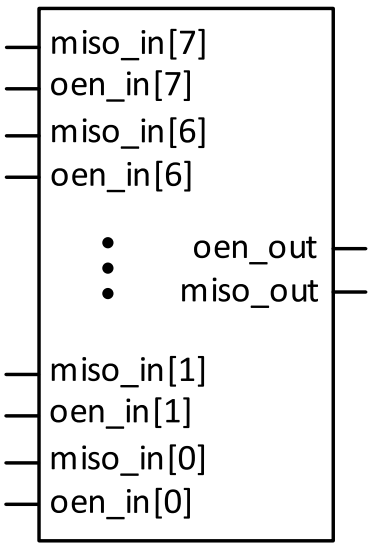

Figure 3: Pin arrangement SPI MUX

Port description

Table 1: Port description SPI slave

Name	Type	Direction	Description	Connection
RST	CMOS	Input	asynchronous reset (low-active)	connect to input pad
ADDR[2:0]	CMOS	Input	Address of macro on SPI bus	input pad or hardwired to Vdd/GND
SCLK	CMOS	Input	SPI clock (low when idle)	connect to input pad
CS	CMOS	Input	SPI chip/slave select (low active)	connect to input pad
MOSI	CMOS	Input	SPI data in (from SPI master)	connect to input pad
MISO	CMOS	Output	SPI data out (to SPI master)	connect to output pad or to MUX MISO_IN
MISO_OE	CMOS	Output	MISO's output enable, high when tristate	connect to output enable of MISO's pad cell or MUX OEN_IN
dout2-dout7[7:0]	CMOS	Output	Output register, writeable via SPI	
din0-din1[7:0]	CMOS	Input	Input ports, readable via SPI	

Table 2: Port description SPI MUX

Name	Type	Direction	Description	Connection
OEN_OUT	CMOS	Output	Multiplexed MISO output enable, high when tristate	connect to output enable of MISO's pad cell
MISO_OUT	CMOS	Output	Multiplexed MISO output	connect to output pad
MISO_IN[7:0]	CMOS	Input	MISO inputs from SPI slaves	connect to miso pin of SPI slave 0-7
OEN_IN[7:0]	CMOS	Input	MISO output enable input from SPI slaves	connect to miso_oe pin of SPI slave 0-7

Logic level at all CMOS ports is 1.2 V (= core supply voltage of SPI block).

Unused output ports doutX, oen_out, miso_out shall be left unconnected.

Unused input ports dinX shall be connected to ground. **Unused input ports miso_in and oen_in must be connected to logic high (1.2 V)!**

Port miso_oe is inactive (= high) while cs = high (inactive) and after the address bits when the address in the SPI word does not match the address set at port "addr" for that block.

IO pads and multiplexer usage

To allow more degrees of freedom during floorplanning, no IO pads are included in the digital cores.

Input pads should be either *I16X* from *ixc013* pad library or *B16M* with output enable port *OEN* connected to logic high. Integrated pullup is recommended for *RST* and *CS*, but optional.

Output pad should be *B16M* from *ixc013* pad library with output enable port *OEN* connected to logic low or to *miso_oe* from SPI block.

SPI block's address input *addr* should be set to "000" (or any other address) by hard-wiring its input ports to ground (or core supply voltage, 1.2 V, respectively). The address input can also be routed to input pads, so that the address can be specified on board level, e.g. with pull-up / pull-down resistors.

Due to the addressable SPI feature, several SPI slaves integrated in separate ICs can share the same SPI bus with the same *CS* and *MISO* lines, if they are configured with different addresses.

If several SPI slaves are integrated in one IC, they can also share the same SPI bus, but the **MISO and MISO_OE output pins must not be directly connected**! To share the same MISO output pad among different SPI slaves, the SPI multiplexer must be used. The following diagrams show different configurations.

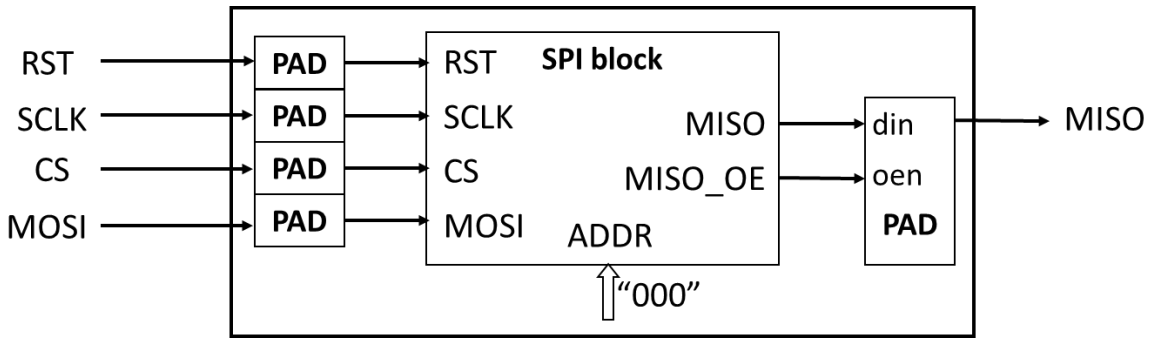

Figure 4: Pad connection using one SPI block on one IC

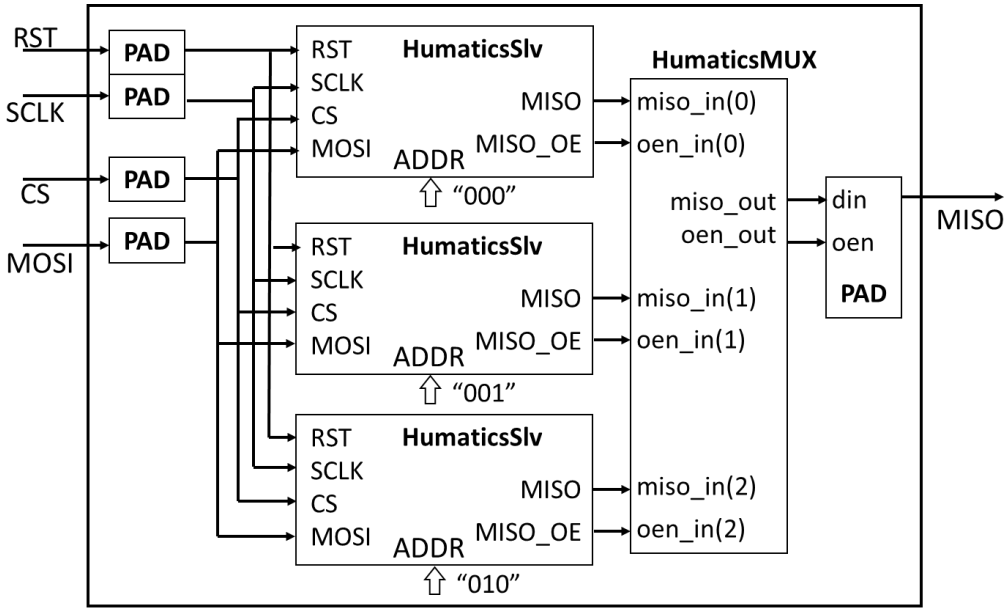

Figure 5: Pad connection using more than 1 SPI block on the same IC *(unused miso_in[..] and oen_in[..] inputs must be connected to logic high = 1.2 V!)*

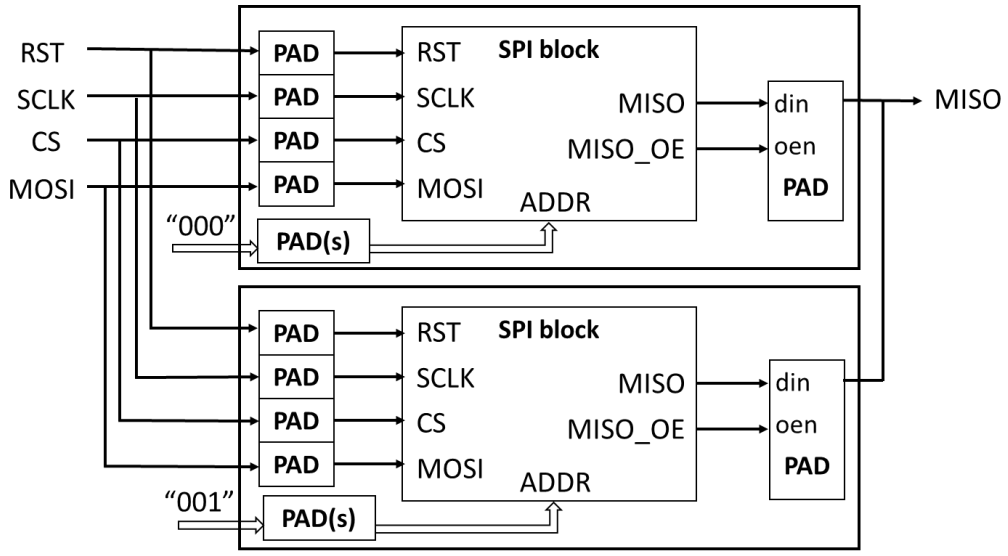

Figure 6: Pad connection with 2 SPI blocks on 2 different ICs

Application hints

We recommend to insert a series resistor of 30 – 50 Ohm into the external SCLK line (cable outside the pad) in order to prevent signal reflections that would distort the SPI operation.

Trouble Shooting

When testing an ASIC containing one of IHPs SPI slave macros, we propose to proceed in the following way:

- Mount the ASIC on a suitable PCB having appropriate power supply for core and IO supply voltages and an SPI connector. Prepare a suitable SPI master (see below) and a PC running some test software. Check for correct logic level on SPI lines (normally 3.3 V).
- Switch on the power supplies and check current consumption on both, core and IO supply voltages. Both values should normally be below 1 mA, if no other parts of the circuit are supplied from these sources (pay attention to pullup/pulldown resistors).

  If current consumption is larger, check for shorts on PCB or perhaps within the ASIC.
- Connect the SPI master to the slave. Do not forget ground connection.

  Perform reset, then read any read/write (i.e. output) register from the slave. This should return the reset/default value of the register. If nothing else is specified for the SPI core, the reset value of SPI registers is zero. If your SPI slave provides at least one register having a non-zero reset value, try to read this register and to confirm its correct reset state. Read-only (i.e. input) registers might have any value after reset according to their connectivity.
- Write a suitable register with some value and read it back. Verify that the value read back is correct. Try this with several different registers and/or values.

  If the returned value is not correct, you may perhaps try to change some register, which has a measurable effect on your chip (e.g. is expected to change the power consumption or the frequency of some synthesizer). Check whether the expected behavior is indeed observed. This allows to see whether SPI writing is correct and only SPI reading does perhaps fail, or whether even SPI write is defective.
- If SPI could not be set into operation, connect an Oscilloscope to the SPI bus. It shall have high impedance inputs (≥ 1 MOhms, max. 20 pF). Normally an operating frequency of 100 MHz is sufficient. Four inputs would allow connecting all 4 SPI lines at the same time. Check the logic level on all SPI lines after reset:

  Reset and SPI chip select shall be on logic high.

  SPI clock shall be on logic low.

  MISO shall have the same level as MOSI. If you can change the input level at MOSI, try this and check, whether the output at MISO follows the input at MOSI.

If this does not work, the MISO pad might be not properly connected (wrong or missing connection of Output Enable signal, insufficient driver strength, problems with ESD diodes, etc.).

- Write some SPI register (having a non-zero address) with non-zero data from your test software and take a snapshot on the oscilloscope (trigger on falling edge of chip select signal CS or rising edge of SPI clock SCLK).

  You should first see CS going down, then a number of clock pulse on SCLK that equals the number of bits in the SPI word, and then CS going up again. During all address bits as well as the read enable bit and the write enable bit (at the end of the word), the MISO output shall be equal to the MOSI input (perhaps with a few ns of delay).

  While data bits are transferred, MISO and MOSI may differ. On MISO one should see the old value contained in the addressed register (even during write operations). Directly after reset, this is normally all 0.

- If MISO output changes slowly (no sharp edge), the driver strength of the pad might be insufficient. If you cannot reduce the load (fan-in) on that wire, perhaps the SPI speed must be reduced.

  If the logic level at MISO does not saturate at the desired voltage (normally ≈ 3 V for logic high and < 0.2 V for logic low), there is either a problem with a second driver on that line (wrong cable connection) or insufficient power supply, insufficient driver strength or improper voltage shifting from core to IO voltage in the MISO pad cell. The latter would require a redesign of the ASIC.

- A common problem is signal integrity on the SPI lines, in particular on the SPI clock SCLK. The SPI design is quite fast, thus it might respond to any spike, overshoot or crosstalk on the clock. The problem becomes more severe with increasing SPI cable length.

  A hint for this kind of problem is instable behaviour of the SPI slave. If the MISO output during the first bits of the SPI transfer (address bits) does not equal the MOSI input, you are probably faced with signal integrity issues.

  In many cases the problem can be solved by inserting series resistors of 30 – 100 Ohms into the SCLK clock line or even into all SPI lines. Try the best value. The resistors should not be too large; signal edges shall be steep enough so that no oscillation may emerge when the forbidden gap between logic high and logic low levels is passed too slowly (otherwise, perhaps Schmitt trigger input pads must be used). A series resistor of 1 kOhm is normally too large.

  Do also check for good ground connection between SPI master and slave.

Revision History

Date	Version	Description of Revision
09/08/2021	V1.0	Initial Version
11/08/2021	V1.1	Document revised

Notice of Disclaimer

IHP Solutions is providing this Intellectual Property, hereinafter “IP”, to you “AS IS” with no warranty of any kind, express or implied. IHP Solutions makes no representation that the IP, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the IP.

All specifications are subject to change without notice, at any time. IHP Solutions assumes no obligation to correct any errors contained herein or to advise you of any correction if such be made. IHP Solutions will not assume any liability for the accuracy or correctness of the provided IP.

IN NO EVENT WILL IHP SOLUTIONS BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOST DATA AND LOST PROFITS, ARISING FROM OR RELATING TO YOUR USE OF THE IP, EVEN IF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

IHP SOLUTIONS EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE IP OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.