SpartanMC

I2C Master

Table of Contents

1. Physical Connection 2

2. Bus Arbitration 2

3. Framing 3

4. Modul parameters 4

5. Peripheral Registers 5

5.1. I2C Register Description 5

5.2. CONTROL Register 5

5.3. TX Register 6

5.4. RX Register 6

5.5. COMMAND Register 6

5.6. STATUS Register 7

List of Figures

1 I2C block diagram 1
2 I2C Pyhsical Connection 2
3 I2C Arbitration 3
4 SCL, SDA Timing for the First Byte 3
5 SCL, SDA Timing for Data Transmission 3
6 I2C Receive Date from Save and Send Data to Slave 4

List of Tables

6 I2C modul parameters 4
6 I2C registers 5
6 I2C control register layout 5
6 I2C transmit data register layout 6
6 I2C receive data register layout 6
6 I2C command register layout 6
6 I2C status register layout 7

I2C Master

The SpartanMC I2C master controller provides a logical connection to a I2C bus. To ensure a proper physical connection it requires pull up resistors on both signal lines (s. physical connection section).

I2C (Inter-Integrated Circuit) also referred to as "Two-Wire Interface" is a single-ended multi-master bus which is used to attach low-speed peripherals to other electronic devices. Originally, the bus system was invented by Philips Semiconductur in 1982. Today, the implementation of the I2C protocol itself is free, but the global assignment of I2C Slave addresses by NXP still requires a fee. A detailed description of the I2C protocol and its license condition could be found on "http://www.nxp.com" or "http://www.nxp.com/documents/user_manual/UM10204.pdf".

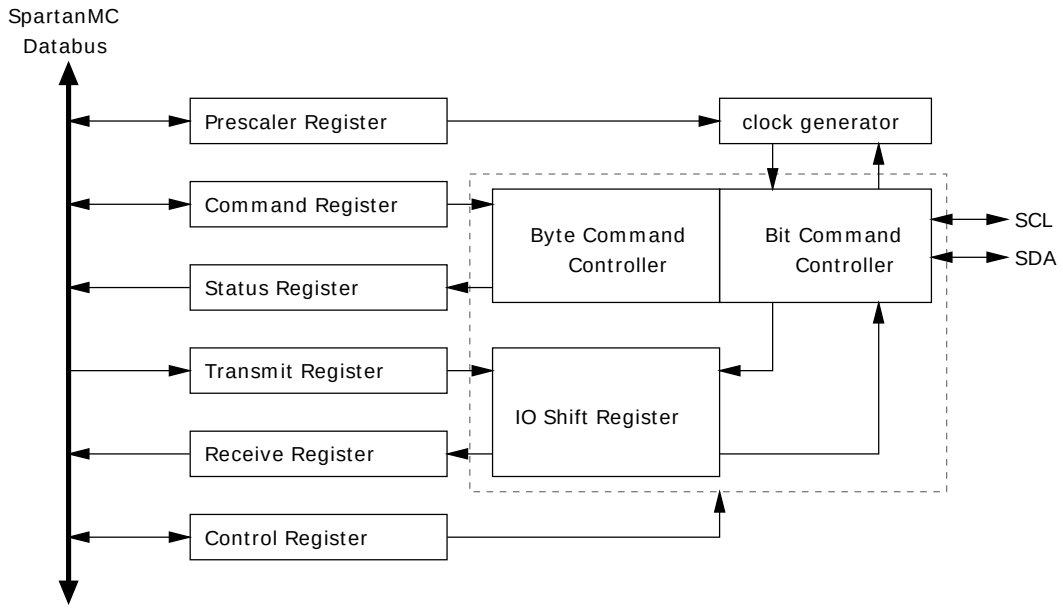

Figure 1: I2C block diagram

1. Physical Connection

The I2C bus requires two signal lines: Seriel Clock Line (SCL) and Serial Data Line (SDA). SDA is used for data transmission while the clock signal SCL is required to synchronize master and slave. The clock signal is always generated by the I2C master. Both lines are connected to VCC via pull up resistors. Thus, a logical zero is generated be pulling the line to ground while a logical one is generated by letting the line float.

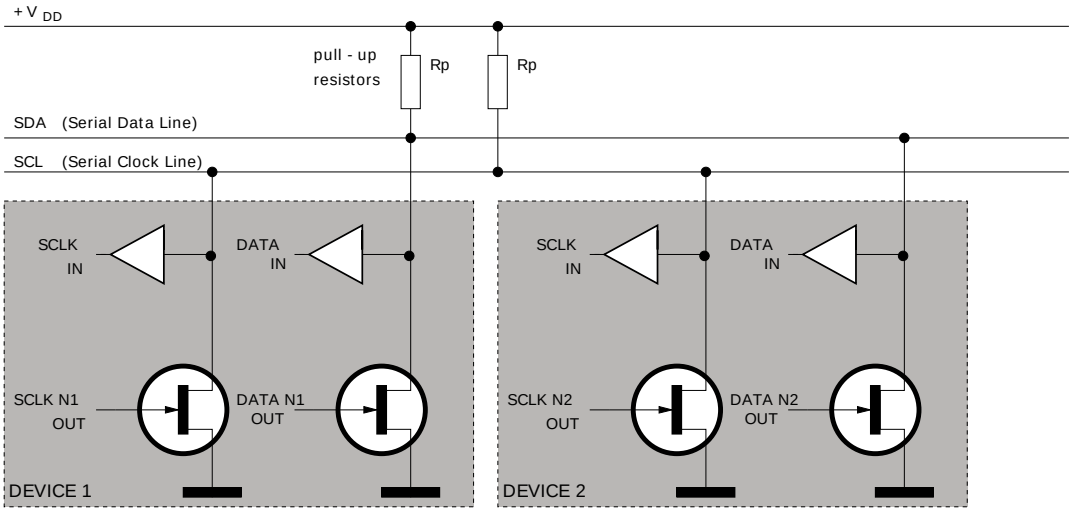

Figure 2: I2C Pyhsical Connection

2. Bus Arbitration

The physical connection mechanism enables a dominant bus level (logical zero) and rezessive bus level (logical one). This could be used for bus arbitration. If different nodes try to drive SDA simultaniously, the node which sends the first zero dominates the bus. The same technique can be used on SCL to give the slave nodes a flow control mechanism. Thus, the data transfer to the master will be delayed when pulling the clock line to zero.

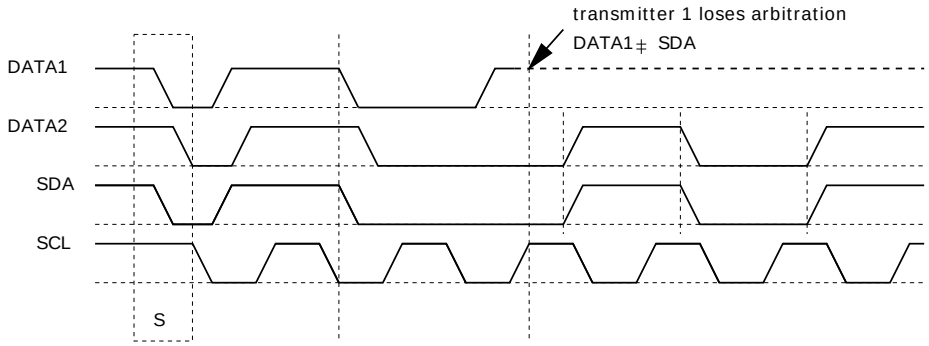

Figure 3: I2C Arbitration

The I2C controller will lost the arbitration when the controller drives SDA high, but the received SDA is low or the controller detects a stop sequence, but non requested one.

3. Framing

To start a transmission, SDA is pulled low while SCL remains high. Each following bit, after this start sequenz, is transmitted with a raising edge of SCL. After transmission SDA must be released to float high again which is used as stop bit and idle marker. Except for the start and stop marker, the SDA line only changes while SCL is low.

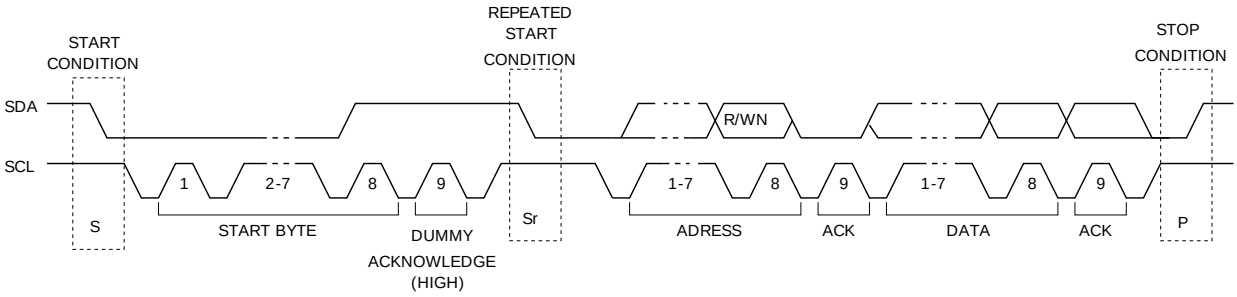

Figure 4: SCL, SDA Timing for the First Byte

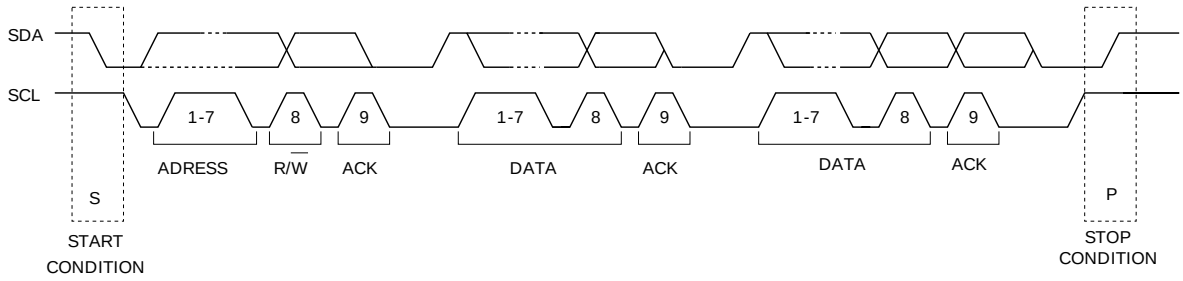

Figure 5: SCL, SDA Timing for Data Transmission

The data transfer is always initilzed by the master. After the start signal the master sends the first byte of the slave address (1 byte for a 7 bit address - 2 byte for a 10 bit address) and a direction bit (1 for read from slave, 0 for write to slave). During data transmission each data byte must be acknowledged throuth the master or slave by pulling SDA to zero.

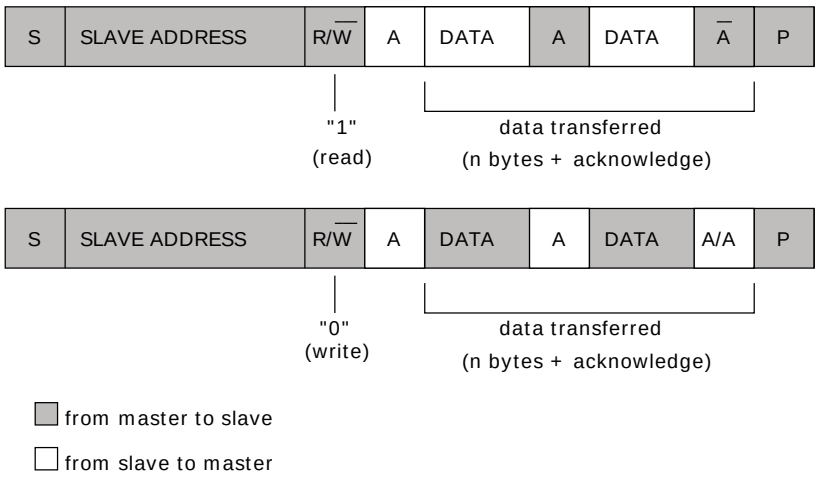

Figure 6: I2C Receive Date from Save and Send Data to Slave

4. Modul parameters

The I2C controller enables a variable registers address offset. Changing the offset to the peripheral base address could be done via following synthesis parameters.

Table 1: I2C modul parameters

Parameter	Default Value	Descripton
CONTROL	0	Address offset for the control register.
TX	1	Address offset for the tx register.
RX	2	Address offset for the rx register.
COMMAND	3	Address offset for the command register.
STATUS	4	Address offset for the status register.

5. Peripheral Registers

5.1. I2C Register Description

The I2C peripheral provides six 18 bit registers which are mapped to the SpartanMC address space. The address offset for the registers refers to to given synthesis parameter (s. module parameter section)

Table 2: I2C registers

Offset	Name	Access	Description
0	CONTROL	read/write	Contains a 16 bit clock divider for SCL gerenration and is used to set and reset the control bits of the I2C controller.
1	TX	write	Contains the current byte to send.
2	RX	read	Contains the current recieved byte.
3	COMMAND	write	Used to set I2C commandos.
4	STATUS	read	Contains the controller status flags.

5.2. CONTROL Register

Table 3: I2C control register layout

Bit	Name	Access	Default	Description
0-15	PRESCALER	read/ write	65535	This register is used to prescale the clock frequency of the SCL line. To change the prescale register the CORE_EN bit must be set to zero. The prescale factor can be dermined through following equation: prescale = (peripheral_clock / (5 * desired_SCL)) -1.
16	CORE_EN	read/ write	0	Enable I2C core. If set to 1 the I2C core is enabled. (The presacler value remains constant.)
17	IEN	read/ write	0	Interrupt enable. If set to 1 the interrupt is enabled.

5.3. TX Register

Table 4: I2C transmit data register layout

Bit	Name	Access	Default	Description
0-7	TX	write	0	Register for data to send. The LSB (Bit 0) of this register represets the read/write bit (1 for read from slave, 0 for write to slave).
8-17	x	write	0	Not used.

5.4. RX Register

Table 5: I2C receive data register layout

Bit	Name	Access	Default	Description
0-7	RX	read	0	Register for received data.
8-17	x	read	0	Not used. (Read as zero)

5.5. COMMAND Register

Table 6: I2C command register layout

Bit	Name	Access	Default	Description
0	IACK	read/ write	0	Interrupt achknowledge. If set to one the pending interrupt will be cleared.
1-2	x	read/ write	0	Not used.
3	ACK	read/ write	0	If set to 0 the acknowledgement of a received frame will be carried out (ACK bit in I2C frame = 0). When set to 1 the I2C frame will be not acknowledged (ACK bit in frame = 1).
4	WR	read/ write	0	If set to 1 the TX register will be written to slave.
5	RD	read/ write	0	If set to 1 the RX register will be filled with data from slave.
6	STO	read/ write	0	Sends stop sequence.
7	STA	read/ write	0	Sends (re-)start sequence.
8-17	x	read/ write	0	Not used.

Note: In case the values for WR and RD are set to 1 simultaneously the controller will set the read bit within the I2C frame.

5.6. STATUS Register

Table 7: I2C status register layout

Bit	Name	Access	Default	Description
0	IF	read	0	This bit is set to 1 when an interrupt is pending and IEN in I2C control register is set. Interrupts are set on the following conditions: • A byte transfer has been completed. • The arbitration is lost.
1	TIP	read	0	Is set to 1 when an I2C data transfer is in progress.
2-4	x	read	0	Not used.
5	AL	read	0	Is set to 1 after an arbitration lost. See arbitration section for a detailed description of arbitration lost conditions.
6	I2C_BUSY	read	0	Is set to 1 after a frame start sequence is detected and set to 0 when the stop sequence occures.
7	RX_ACK	read	0	Is set to 1 if the acknowledgement from slave failed.
8-17	x	read	0	Not used.