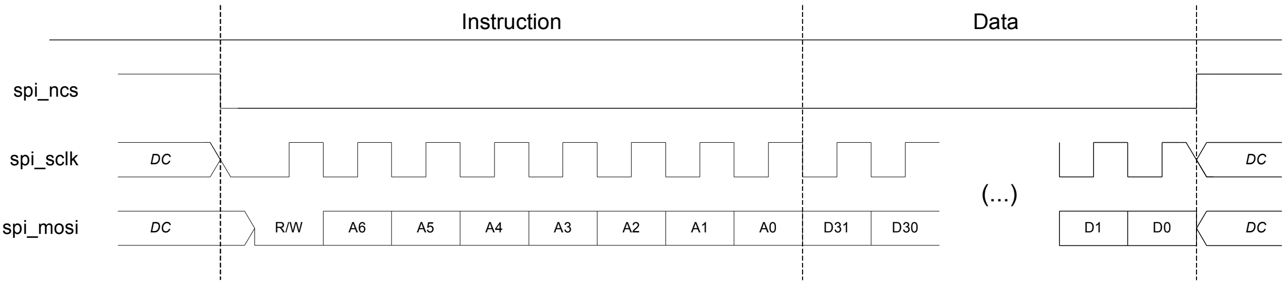
SPI configuration port

A write or read operation is initiated by pulling **spi_ncs** low. There are two parts to a communication cycle with the FPGA. In the first part, a 8-bit instruction word is written to the FPGA, coincident with the first 8 **spi_sclk** rising edges. The instruction word provides information regarding the data transfer, which is the second part of the communication cycle. The instruction word defines whether the upcoming data transfer is a read or a write and the starting register address for the first word of the data transfer.

7	6	5	4	3	2	1	0
R/$\overline{W}$	ADDR						

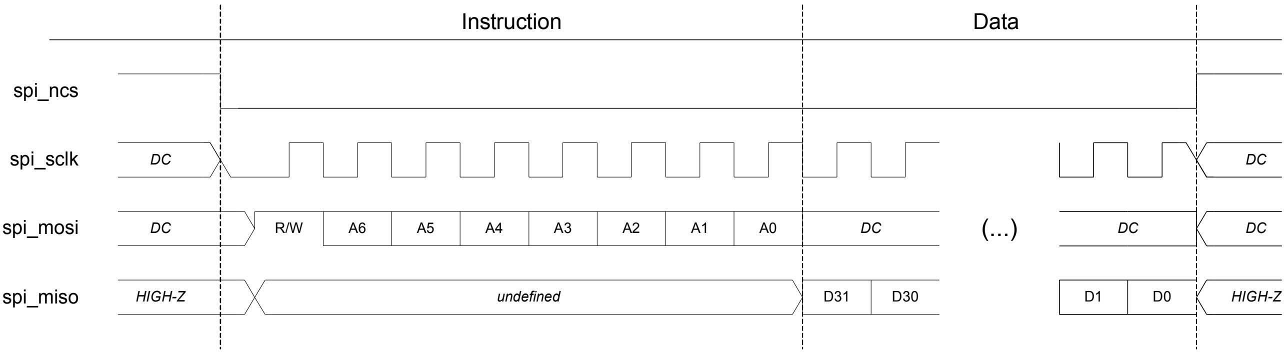
Setting a register value

If the instruction word is for a write operation, the next 32 **spi_sclk** cycles clock in data for the address specified in the instruction word. Data bits are registered on the rising edge of **spi_sclk**.



Reading a register value

If the instruction word is for a read operation, the next 32 SCLK cycles clock out the data from the address specified in the instruction word. The readback data is updated on the falling edge of **spi_sclk**.



Register set description

Register 0x00

Identification

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

MAGIC	[31..0]	R	**Magic Identifier**

32 bit unsigned integer

The magic value can be used to detect is the FPGA is running and the communication interface is working. It always reads **0xDEADBEEF** at the moment.

Register 0x01

PWM #1

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

FREQ [15..0] R/W **PWM1 Frequency**

16 bit unsigned integer

The frequency of the first PWM output is configured vis this value:

$$PWM_{freq} = \left(\frac{80\,MHz}{FREQ+1} \right)$$

DUTY [31..0] R/W **PWM1 Duty Cycle**

16 bit unsigned integer

The duty cycle of the first PWM output is configured vis this value:

$$PWM_{duty} = \left(\frac{DUTY}{FREQ+1} \right)$$

Register 0x02

PWM #2

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="16"><b>DUTY</b></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16"><b>FREQ</b></td></tr>
</table>

FREQ [15..0] R/W **PWM1 Frequency**

16 bit unsigned integer

The frequency of the second PWM output is configured vis this value:

$$PWM_{freq} = \left(\frac{80\,MHz}{FREQ+1} \right)$$

DUTY [31..0] R/W **PWM1 Duty Cycle**

16 bit unsigned integer

The duty cycle of the second PWM output is configured vis this value:

$$PWM_{duty} = \left(\frac{DUTY}{FREQ+1} \right)$$

Register 0x03

ADC Timing

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

CLKDIV [7..0] R/W **AD7357 SCK Frequency**

8 bit unsigned integer

The frequency of the SCK line to the AD7357 ADC is configured via this value:

$$F_{SCK}=\left(\frac{80\,MHz}{CLKDIV}\right)$$

DUTY [15..8] R/W **AD7357 Acquisition Time**

8 bit unsigned integer

Number of SCK cycles the CS Line to the AD7357 is held high between two conversions.

Register 0x04

SSC Interface

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

TM [0] R/W **Testmode**

Decides if the testmode of the SSC Interface is enabled:

0	normal operation
1	testmode enabled, data is replaced by a 16 bit counter

CLKDIV [15..8] R/W **SSC Clock Frequency**

8 bit unsigned integer

The frequency of the Clock line of the SSC interface to the MCU is configured via this value:

$$F_{SSC}=\left(\frac{80\,MHz}{2\cdot(CLKDIV+1)}\right)$$

Register 0x06

Decimation Filter

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

RATE [2..0] R/W **Decimation Factor**

Configures the decimation factor of the filter:

000	no decimation
001	Samplerate is reduced by factor **2**
010	Samplerate is reduced by factor **4**
011	Samplerate is reduced by factor **8**
100	Samplerate is reduced by factor **16**
101	Samplerate is reduced by factor **32**
110	Samplerate is reduced by factor **64**

Register 0x09

ADC Channel Swap

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

SWAP [0] R/W **Channel Swap**

Decides if the ADC channels get swapped:

0	normal operation	$I_{swap} = \mathrm{I}$ $Q_{swap} = \mathrm{Q}$
1	inverted spectrum	$I_{swap} = \mathrm{Q}$ $Q_{swap} = \mathrm{I}$

Register 0x08

ADC Sample Gain

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

GAIN_I [15..0] R/W **Gain (I-Channel)**

16 bit unsigned integer

Each **I** sample is scaled with the value of **GAIN_I**:

$$I_{scaled} = \left(\frac{I_{swap} \cdot \mathrm{GAIN\_I}}{32768} \right)$$

GAIN_Q [31..0] R/W **Gain (Q-Channel)**

16 bit unsigned integer

Each **Q** sample is scaled with the value of **GAIN_Q**:

$$Q_{scaled} = \left(\frac{Q_{swap} \cdot \mathrm{GAIN\_Q}}{32768} \right)$$

Register 0x07

ADC Sample Offset

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

OFF_I [15..0] R/W **Offset (I-Channel)**

16 bit signed integer

The value off **OFF_I** is added to each I sample:

$$I_{out} = I_{scaled} + \mathrm{OFF\_I}$$

OFF_Q [31..0] R/W **Offset (Q-Channel)**

16 bit signed integer

The value off **OFF_Q** is added to each Q sample:

$$Q_{out} = Q_{scaled} + \mathrm{OFF\_Q}$$

Register 0x0A - 0x0C

0x0A - GPIO – Output Enable

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					GPIO_OE										

0x0B - GPIO – Output Data

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					GPIO_OD										

0x0C - GPIO – Input Data

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					GPIO_ID										

GPIO_OE [10..0] R/W **Output Enable**

Sets the direction of the GPIO Pin:

0	GPIO is input
1	GPIO is output

GPIO_OD [10..0] R/W **Output Data**

Sets the data written on the Output driver:
(only relevant if the corresponding **GPIO_OE** Bit is set)

0	GPIO is driven low
1	GPIO is driven high

GPIO_ID [10..0] R **Input Data**

Gets the data read from the Input driver:

0	GPIO is low
1	GPIO is high

GPIO Mapping:

- GPIO[0:9]: A[1-10] Pin
- GPIO[10]: FPGA LED

Register 0x0D

Reference Frequency Measurement

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

REF_CNT [24..0] R **Reference Counter**

Return the number of the **CLKPO** (typical 30MHz) Cycles between the last two rising edges on the **1PPS** Signal.

REF_UPD [31:25] R **Reference Update Counter**

This 7 Bit Counter is incremented every time a new value is latched into the REF_CNT register.