**Mud sensor FPGA**

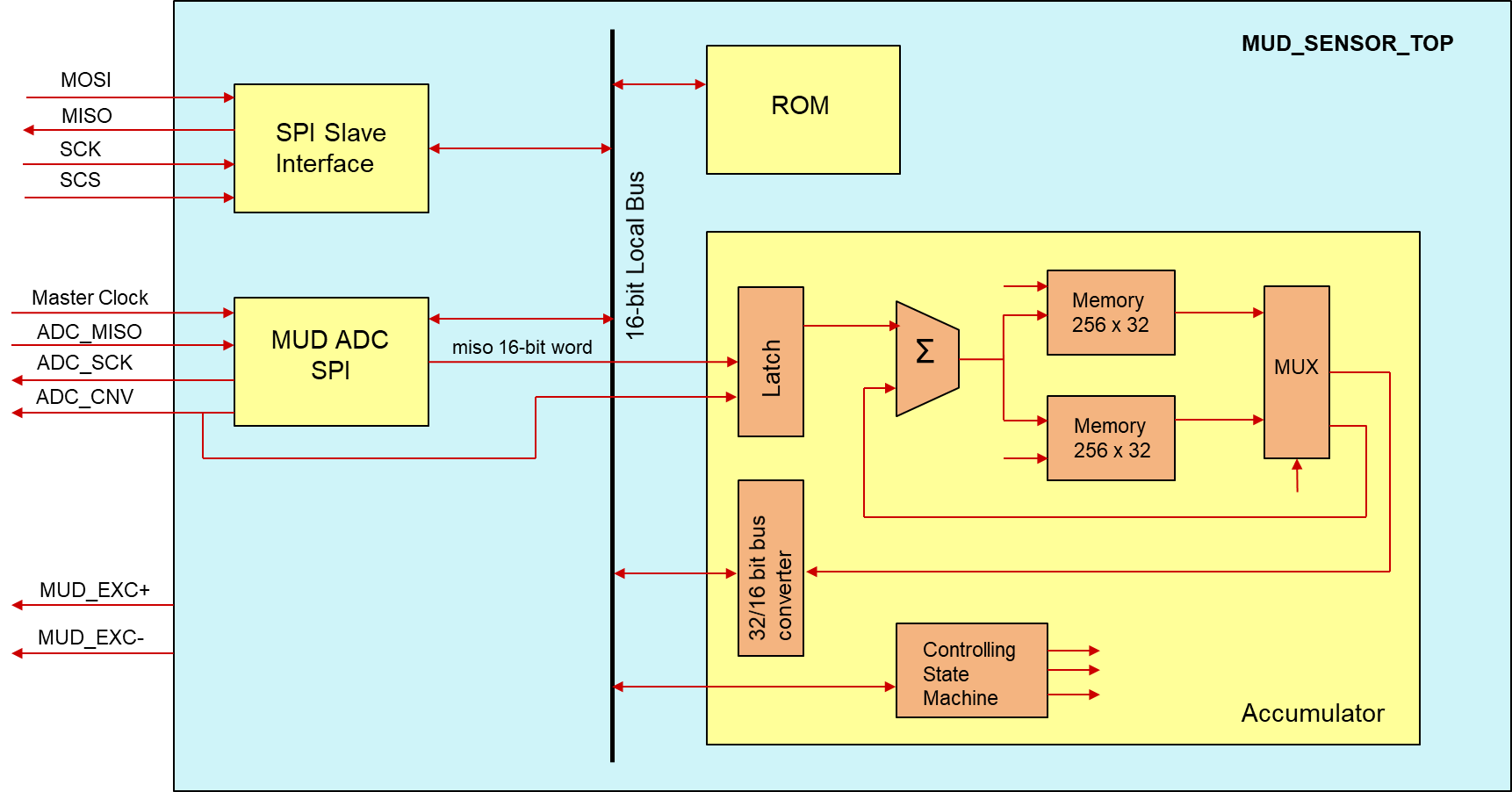
***Introduction***

This document describes the FPGA implementation of mud sensor for NG Resistivity project. NG TX board supports the mud sensor functionality. FPGA used on this board is Actel AGL250V2 (P/N 102512867) which facilitates data acquisition and accumulation for mud sensor. It connects to the NG RX board processor via standard 4-wire SPI interface.

FPGA implementation of mud sensor is inspired from XHTADR Receiver FPGA implementation. Few other comments on mud sensor implementation:

1. Mud sensor excitation signal (20kHz) is provided by FPGA.
2. Read-accumulate-write operations of FPGA are synced with ADC CNV clock (160kHz).
3. Two 256x32 SRAM blocks are created to support simultaneous read and accumulate-write FPGA operations.

***FPGA Block Diagram***

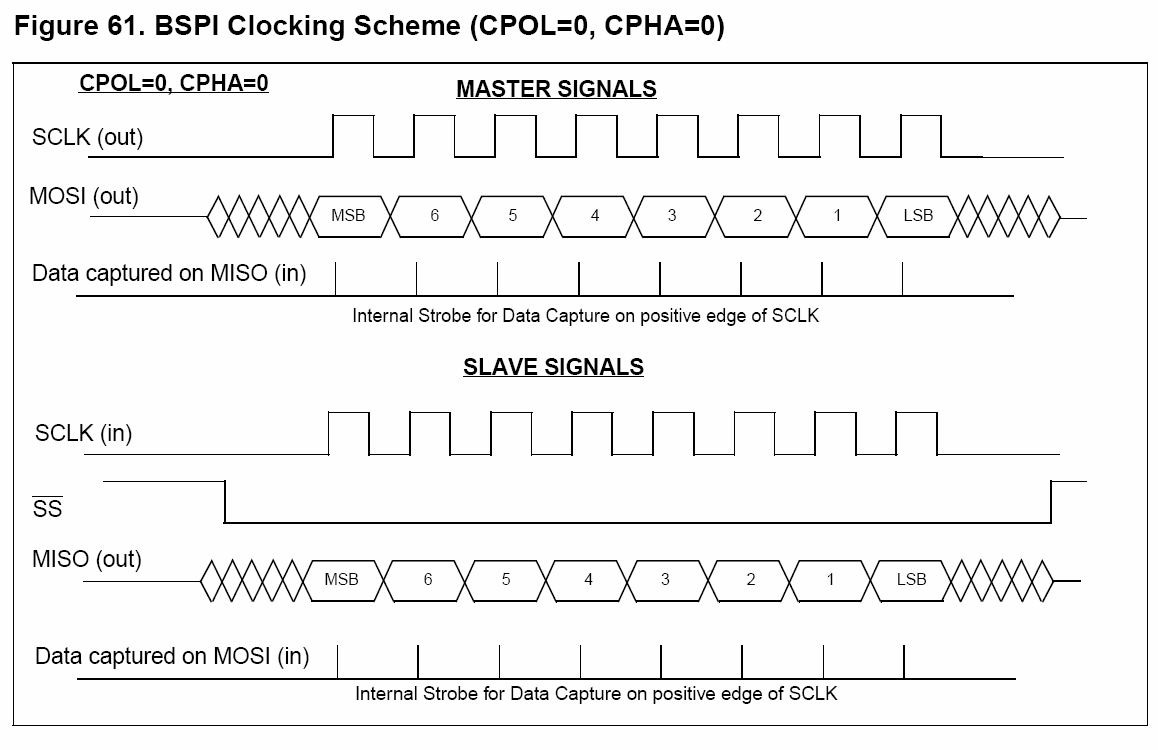


***FPGA Compile-Time Configuration Parameters***

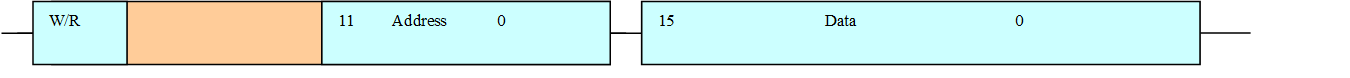
Most of the configurable FPGA parameters are defined in package conf\_pkg.vhd. This file is to be used as a definitive reference for memory map, bit definitions, etc.

***SPI Communication Interface***

The frame of the SPI data exchange is usually described by two parameters, the clock polarity (CPOL) and the clock phase (CPHA). These parameters define 4 possible SPI modes (0-3). Our implementation configures SPI interface to run in mode 0. Other modes have not been tested.



Each SPI transfer consists of two 16-bit words. The first word contains command and address and the second word is data:



W/R bit defines if we are to execute write (1) or read (0) command. It is followed by 3 don’t care bits and 12 address bits. Data is always transferred as a 16-bit word. If a peripheral does not make use of all 16 bits, then unused bits are “don’t care” on write & “undefined” on read.

*MCj03073590000[1]Warning: user software must mask out undefined bits on read in order to obtain correct value.*

***Memory Map***

SPI to parallel bus abstraction layer allows us to describe the access of internal peripherals in terms of memory map as given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Int. Peripheral** | **Start Address** | **End Address** | **Configuration** |
| ROM | 0x020 | 0x020 | Read only |
| Mud ADC SPI | 0x021 | 0x021 | Read only |
| Mud Input Select | 0x022 | 0x022 | Read/Write |
| Mud Cal | 0x023 | 0x023 | Read/Write |
| Acc. Control | 0x040 | 0x042 | Read/Write |
| Acc. Memory | 0x400 | 0x5FF | Read only |

*Note: Register maps for individual peripherals are described in terms of offset from the peripheral base (start address). For example, samples per acc. register’s absolute location can be calculated as (acc. control base + samples per acc offset) = 64 + 1 = 65 (0x0041).*

***ROM (temporary/for debugging)***

ROM is a 1 byte read only register used for storing the FPGA rev. number. The content of this register is defined in conf\_pkg.vhd VHDL package (see ROM\_REVISION\_MUD and SUB\_REVISION\_MUD).

***Mud ADC SPI (temporary/for debugging)***

This module acts as SPI master for communication with mud ADC. It provides 4MHz SPI clock, 160kHz sampling clock to ADC and reads ADC miso line. Reading this module outputs 16-bit ADC data.

***Mud Input Select***

This is a 1-bit register that selects input, MUD\_V or MUD\_I, for ADC sampling. Setting this register ‘1’ selects *MUD\_V* and ‘0’ selects *MUD\_I*.

***Mud Cal***

This is a 1-bit register that selects between the following modes: Mud Measurement and Calibration. Setting this register ‘0’ selects *Mud meas*. and ‘1’ selects *Cal.*

***Acc. Control***

This configures three registers for FPGA data accumulation. See register map below.

*Register map:*

|  |  |  |  |
| --- | --- | --- | --- |
| **Register Offset** | **Register Width** | **Description** | **Default Value** |
| 0x0 | 8 | No. of accumulators. | 0x07 |
| 0x1 | 16 | No. of samples per accumulator | 0x0005 |
| 0x2 | 10 | Acc. Status | 0x0040 |

Acc. Status

|  |  |  |
| --- | --- | --- |
| Bit | Description | Function |
| 0 | don’t care | X |
| 1 | don’t care | X |
| 2 | don’t care | X |
| 3 | start bit (wo) | ‘1’ – start acquisition  ‘0’ – does nothing |
| 4 | busy bit (ro) | ‘1’ – acc. busy  ‘0’ – acc. idle |
| 5 | cont bit (rw) | ‘1’ – continuous acquisition  ‘0’ – snapshot acquisition/stop continuous acquisition |
| 6 | read memno bit (ro) | ‘1’ – mem 1 available for read  ‘0’ – mem 0 available for read |
| 7 | mem overwrite status bit (ro) | ‘1’ – mem0 overwritten at the time of read  ‘0’ – mem0 not overwritten |
| 8 | ‘1’ – mem1 overwritten at the time of read  ‘0’ – mem1 not overwritten |
| 9-15 | don’t care | X |

rw – read/write; ro – read only

***Acc. Memory***

This is SRAM for data accumulation. Reading SRAM returns 16-bit data value stored at the memory address specified by the user. Since memory is 32-bits wide and SPI data frame is 16-bits, a 32-bit to 16-bit converter is used (see FPGA block diagram). Two read operations need to be performed to retrieve data stored in one accumulator. As an example, shown below is a 32-bit wide SRAM with 8 accumulators. Hex numbers in grey boxes indicate register offsets to access data in all 8 accumulators.

16-bits 16-bits

|  |  |  |
| --- | --- | --- |
| Acc 0 | 0x1 | 0x0 |
| Acc 1 | 0x3 | 0x2 |
| Acc 2 | 0x5 | 0x4 |
| Acc 3 | 0x7 | 0x6 |
| Acc 4 | 0x9 | 0x8 |
| Acc 5 | 0xB | 0xA |
| Acc 6 | 0xD | 0xC |
| Acc 7 | 0xF | 0xE |

32 bits

**Data Accumulation Process**

The accumulator stores/accumulates data continuously in real time (if set in continuous mode). 16-bit ADC data is latched into an input register every rising edge of ADC sampling clock. It is then passed onto a 32-bit adder, output of which is accumulated into one of the two memory blocks (mem0 or mem1). The other memory block is available to the user for reading. At the end of the data acquisition cycle, the memory blocks are swapped.

Memory is organized into two blocks of 256x32 accumulators. Each block provides up to 128 accumulators and the last 128 accumulators are unused. The actual data accumulation occurs in the first N accumulators, where is N is programmable from 1 to 127 (Nprogrammed = Nactual-1). Also, samples per accumulator are configurable by user and can be set from 1 to 216-1 (Nprogrammed = Nactual). Thus, total no. of samples are calculated as: *no. of acc.\* samples per acc.*

First sample for each accumulator is stored and the rest are accumulated. For example, if total no. of accumulators are set as 7 and samples per accumulator are set as 5 (total samples: (7+1)\*5 = 40), first sample will be stored in each accumulator and the remaining samples will be cumulatively added. Sampling to signal frequency ratio must be chosen to ensure that each accumulator receives the samples of same signal phase. Current setting for signal and sampling frequency is 20kHz and 160kHz respectively.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sample no. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | N |
| Acc. no. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | N%8 |
| Sig. phase | 0 | 45 | 90 | 135 | 180 | 225 | 270 | 315 | 0 | 45 | 90 | 135 | 180 | 225 | N\*45 |