|  |
| --- |
|  |
| TSW1400 ADC: Firmware/ DLL revamp  |
| FPGA Design Specification & Firmware Guide |
|  |
| **Prakash Mathews Pothen** |
|  |

|  |
| --- |
| Soliton Logo |

 **Revision 0.3**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Overview**The Stratix VI FPGA has 4 differential I/O banks bank 1, 2, 5 and 6. From these banks **38** LVDS channels are routed to the Samtec connector. Out of which **2** are dedicated clocks to Left and right side I/O banks of the FPGA. Refer **Table 1** for the number of channels and clocks available in each bank. Refer the TSW1400-ADC interface guidelines document for designing a new ADC interface board for TSW1400.**Table 1: Channel and clock distribution across Banks.**

|  |  |  |
| --- | --- | --- |
| **FPGA device Side >>** | **Right side** | **Left side** |
| **FPGA I/O Bank >>** | **Bank6** | **Bank5** | **Bank2** | **Bank1** |
| **# of LVDS channels** | 13 | 13 | 5 | 3 |
| **# of High speed clock** | 0 | 1 (AA7) | 1(AA28) | 0 |
| **# of Clock/Data pins** | 1(M6) | 1(AB6) | 0 | 0 |

**Architecture**Refer figure 1 for ADC design block diagram. There are two PLL’s, PLL\_L2 (left side) and PLL\_R2(right side). PLL\_L2 will provide clock to the LVDS channels at the left side of the FPGA while PLL\_R2 will provide clock to the channels at the right side. There are totally 3 LVDS Deserializer modules, one for Bank5 channels, one for Bank6 channels and one for Banks1 &2. The deserialization factor is **four.** There will be a 144 (36 x4) bit output from all the deserializers out of which only some bits will be valid **Figure 1: ADC Module Design Block Diagram****Channel Selection and reordering multiplexers**CH1\_INDX1114 x4bitbus |  | The outputs from the Deserializers are selected based on configuration settings send from GUI. This gives an output of X bits. Where X= number of valid channels multiplied by 4. For example if it is an 8 channel 1 wire ADC the output will be (8x4=32 bits) at one parallel clock. Implies X=32bit and Y=128/32 = 4. So 32 bits of valid data will be selected from the 144 bits output from the 3 Deserializers and four samples of the 32 bit will be shifted to form a 128 bit output.**Channel Selection and reordering**There is a multiplexer for each output channel with an input of 36 possible channels. GUI will send the selection/index for that particular channel. For example if output channel 1 has to be routed to channel-A which is routed to input 13 then configuration register (CHNL1\_INDX) will have the value 13. Hence output 1 will be directly routed to input 13. The channels can be easily reordered using this scheme. Invalid channels will be assigned zero.**Output Buffer**To provide maximum throughput, invalid data should not be written in to the Ram and it should not be send to the GUI. Every time an X bit is generated it is shifted in to a 256 bit register. This will be done many times to accumulate the register. When 128 bits are filled in the register it is written in to the RAM. When the data is not in exact multiples the remaining data is send with the next 128 bit data. Hence throughput is maintained by sending only the valid data.N = number of valid channelsX= Nx 4 |

Bank 5 clock (right)

2

36

PLL\_R2

Bank 5 Deserializer

**Data packing**

**(Shift Register)**

32 channels of 4 bit bus out of which only first N channels are valid

CH2\_INDX

14 channels

 from Bank5

1

2

Left Shift by X bits

2

36

14 x4bitbus

Bank 6 Deserializer

Frame clock check?

Buf

128 bit valid data

Output to RAM

Right Shift by Counter - 128

14 channels

 from Bank6

EN

CH32\_INDX

Counter

2

1

EN

32

Bank 2 clock (left)

36

8 x4bitbus

Bank 1 &2 Deserializer

PLL\_L2

Detect start of Sample

5 channels from bank 2;

3 channels from bank 1

EN

1

FRM\_CLK\_INDX

Pack Frame Clock

2

Frame clock

Frame clock ADC

EN

36

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  **Example : 4 channel 2 wire ADC**Assume that four channels of Wire1 of the ADC is connected to Bank5 and four channels of Wire2 is connected to Bank6. Below table gives the channel inputs going to the Deserializer 1 & 2.**Table 2: Channels at the output of the Deserializer**

|  |  |
| --- | --- |
| **Bank 5** | **Bank 6** |
| **S/N** | **Channel input** | **S/N** | **Channel** |
| 1 | A1 | 16 | A2 |
| 2 | C1 | 17 | Not used |
| 3 | Not used | 18 | D2 |
| 4 | B1 | 19 | Not used |
| 5 | D1 | 20 | B2 |
| 6 | Not used | 21 | Not used |
| 7 | Not used | 22 | Frame clock |
| 8 | Not used | 23 | C2 |
| 9 | “ | 24 | Not used |
|  | “ |  | “ |
|  | “ |  | “ |
| 15 | “ | 29 | “ |

Out of these channels only A1-D1 and A2-D2 are needed. Configuration registers are sent from GUI such that the necessary channels only are used. Channels are reordered according to the wires making it easy for the DLL to decode. Refer table 3.

|  |
| --- |
| **Frame clock** If the frame clock sync bit is high then the frame clock data alone is packed in to the 128 bit buffer and send. Frame clock is also used to detect the start of a valid sample. Frame clock rising edge is considered as the start of a sample. However this is done only for the start of one capture cycle and frame clock is not used for any other purpose  **Ini files** The new ini files should have all the channel indexes, the frame clock indexes, number of channels, ADC with frame clock or not etc. |

 |  | **Table 3: Channels and their configuration registers**

|  |  |  |
| --- | --- | --- |
| **S/N** | **Channel out** | **Config. Reg. value** |
| 0 | Frame Clk | 22 |
| 1 | A1 | 1 |
| 2 | A2 | 16 |
| 3 | B1 | 4 |
| 4 | B2 | 20 |
| 5 | C1 | 2 |
| 6 | C2 | 23 |
| 7 | D1 | 5 |
| 8 | D2 | 18 |
| 9 | Not used | 0 |
| : | “ | “ |
| 31 | Not used | “ |

The final 128 bit output will be of the format given below

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D2 | D1 | C2 | C1 | B2 | B1 | A2 | A1 | T1 {127:96} |
| D2 | D1 | C2 | C1 | B2 | B1 | A2 | A1 | T2 {95:64} |
| D2 | D1 | C2 | C1 | B2 | B1 | A2 | A1 | T3 {63:32} |
| D2 | D1 | C2 | C1 | B2 | B1 | A2 | A1 | T4 {31:0}e final 128 bit output will be of the format given below |

 |

**FPGA Code explanation**

|  |  |
| --- | --- |
| **D:\Projects\TMD_projects\TSW1400\Firmware Ultimatum\Documents\mainproject.bmp**The image shown above shows the project structure. The top module is tsw1400\_top. It interconnects all other modules and has some trigger delay functionality. The important modules are1. adcif: Takes the LVDS data in and gives out deserialized data to RAM
2. ddr2 : DDR2 IP which is a memory controller
3. Dumpmem\_bridge : Sends the ADC data to memory
4. Dumpmem\_config: This module takes care of all the commanding from GUI. It has all the config registers read & write
5. Dumpmem: Reads memory and sends the data to SPI/USB
 | **D:\Projects\TMD_projects\TSW1400\Firmware Ultimatum\Documents\projectexpanded.bmp**Adcif module and dumpmem\_config module will be explained further for these are the only two main modules modified for Firmware/DLL revamp.  |
| **Adcif module**Adcif module consists of the below listed sub modules1. PLL0 : PLL for bank5 ADC clock; clock outputs
* ADC clock x 2 (DDR clock)
* ADC clock/2 (deserialized output data clock)
1. PLL1: Not used (bank 1)
2. LVDS0a: deserializer IP (bank5 14 channels)
3. LVDS0b: deserializer IP (bank6 14 channels)
4. LVDS1 : not used (bank 1 &2)
5. Channel selection and reorder: Multiplexes the input channels for selecting only the valid data and the valid data in the LSB of the 128 bit output
6. Invert\_4bits: reverse each nibble in the 128 bit data.
7. Pack\_data: Valid data is shifted in to the shidt register buffer and when the count exceeds 128 the buffer is shifted to output buffer and then send to memory.
 |  |

 **Revision log**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Modification details** | **Date of release** |
| Rev0.0 | Initial release for discussion | 16th July 2012 |
| Rev0.1 | Modifications made to Architecture after discussions with Anand, Varalakhsmi and Arjun. | 18th July 2012 |
| Rev0.1.1 | Corrected number of channels to 38 and number of bits to 152 bit.  | 20th July 2012 |
| Rev 0.2 | Modified block diagram and description for the latest architecture | 28th Aug 2012 |
| Rev 0.3 | Added code explanation and changed shift register block diagram | 14th Sept 2012 |