Exercise 2

Implement a FIR filter co-processor in FPGA

Design an FIR filter both VHDL and in a programming environment of your choice (i.e. python) to compare the results from the two implementations (i.e. Frequency analysis). The FIR filter could have an arbitrary number of "taps" and an arbitrary behavior in the frequency domain (i.e. Low-pass, high-pass, band-pass, notch). The data width of the samples will be **32**. The size of the data processed by the FIR. Starting from the codebase included in the project "final_ex" delivered in the CernBox, you are requested to build a VHDL project able to run in the Arty7 Board with the following requirements:

1. The Arty7 board is in charge to apply the FIR filter to the data sent by a PC
2. It is possible from a PC (i.e the NUC) to connect to the arty 7 via IPBUS
3. It is possible to send the input data sequence (few hundred of samples) and get the result back using the IPBUS protocol
4. Use the *ipbus dual ram* included in the given design **final_ex** to interface the IPBUS sysntem with the FIR block
5. The data are processed by the FIR filter in FPGA
6. The data results of the FIR operations have to be compared to the same

Compose a technical report that explains the various developement phases, simulations, results.

Hints

Interfaces the FIR filter, which tipically has a *serial* interface (the input data has to be loaded *one sample per clock cycle*), with the blockram; to do that implements a dedicated state machine able to perform *read* and *write* operations (have a look to the figure below). In the case of using python, the **scipy** library has a method *firwin* that helps to calculate the coefficients related to a specified frequency behavior. Take into account that the FIR filters have as many transient states as the number of taps.

https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.firwin.html

Use the Blockram divided into 2 logical sections: the upper half for the inputs and the lower half for the output.

IPBUS example session in python

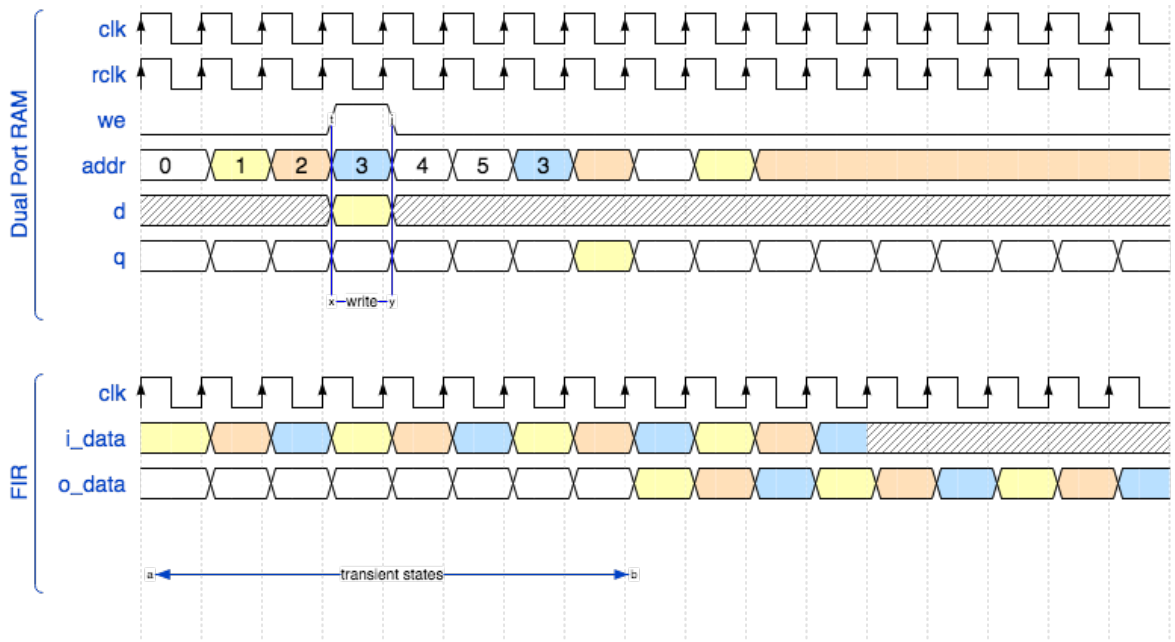
```
$ source ipbus-software/uhal/tests/setup.sh
 # create the environment for the IPbus

$ python -i ipbus_test.py

Python 2.7.16 (default, Apr 12 2019, 15:32:52)
[GCC 4.2.1 Compatible Apple LLVM 10.0.0 (clang-1000.11.45.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
20-01-20 17:12:23.539408 [7f869b5b3740] INFO - Reading XML address file
↪  "/home/antonio/Lab9/Progetto/software/arty7_regs.xml"
20-01-20 17:12:23.539444 [7f869b5b3740] INFO - ConnectionManager created
↪  node tree:
Node "TOP", Address 0x00000000, Module
↪  "/home/antonio/Lab9/Progetto/software/arty7_regs.xml"
Node "regs", INCREMENTAL block, Size 256, Addresses [0x00000000-000000FF],
↪  Permissions rw, Description "flash registers", Module
↪  "/home/antonio/Lab9/Progetto/software/arty7_regs.xml"
20-01-20 17:12:23.539464 [7f869b5b3740] INFO - URI
↪  "ipbusudp-2.0://10.10.10.100:50001" parsed as:
protocol : ipbusudp-2.0
hostname : 10.10.10.100
port : 50001
path :
extension :
arguments :
>>>

>>> data = hw.getNode('regs').readBlock(20)
>>> hw.dispatch()
>>> data.value()
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
>>> data_to_write=range(20)
>>> data = hw.getNode('regs').writeBlock(data_to_write)
>>> hw.dispatch()
>>> data = hw.getNode('regs').readBlock(20)
>>> hw.dispatch()
>>> data.value()
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

Example waveforms



Useful links

1. Arty7 schematic https://reference.digilentinc.com/_media/reference/programmable-logic/arty/arty_sch.pdf
2. IPbus tutorial https://ipbus.web.cern.ch/ipbus/doc/user/html/

for any questions please write to: bergnoli@pd.infn.it