ES6151 PROJECT REPORT

Implementation of signal spectrum analyzer on ZYNQ SoC

CHETHAN KUMAR
AJITH

Abstract

This project aims at software, hardware and co-design of signal spectrum analyzer algorithm on Xilinx ZynqTM-7000. Focus of this part of the project is to use the hardware and software resources effectively to develop an efficient system with minimal software execution time and FPGA resources. To achieve high performance DMA is used to stream data between PS and PL. Execution time achieved for this project (Co-design) is **30us** by utilizing **1071** slice registers and **818** LUT's of FPGA recourses.

1 Introduction

ZYNQ SoC is used as platform to implement the signal spectrum analyzer; it is a versatile architecture which has FPGA + ARM. Depending on the requirement either FPGA can be used or just ignored. FPGA which is high performance computing device is capable of performing at much higher rate compared to general purpose processor like ARM. In this project we focus mainly on how to make use of the available resource effectively to improve the performance of overall system. There are five stages involved in this project audio samples capture, FFT computation, magnitude calculation, moving window averaging and the noise cancellation. Partitioning of tasks is done in such a way that time consuming tasks like FFT and magnitude are offloaded to FPGA. ARM processor takes care of averaging and noise cancellation job.

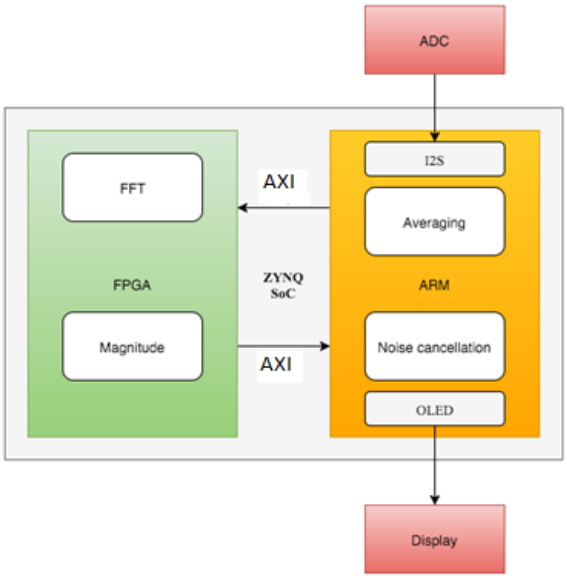


Figure 1 High level design

2 PS

This section explains the modules that are implemented in FPGA. As already mentioned FFT and magnitude module are implemented as part of PS section. FPGA receives audio data from processor through AXI streaming interface, computes magnitude followed by FFT and send back magnitude values to the processor for the further processing.

2.1 FFT

We have used FFT 8.0 Xilinx core to compute FFT. This core is configured for the following parameters.

- Target clock frequency: 100MHz
- Transform length: 128
- Radix-2 Lite burst I/O
- Fixed point
- Unscaled
- Memory: Distributed RAM
- Butterfly Arithmetic: Use CLB logic

FFT 8.0 is used specifically, because, it supports AXI streaming so that we can avoid buffering of data in PS. This in turns improves the effective utilization of PS recourses.

2.2 Magnitude

Magnitude module takes output from the FFT core which consists of both real and imaginary part and computes magnitude. Implementing magnitude module as part of FPGA gives advantage of transferring only 128 data after computation of magnitude, otherwise we will end up in sending both real and imaginary part of FFT output which is quiet expensive in terms of time.

3 PL

In this section we explain the implementation details of averaging and noise cancellation modules in ARM processor. Averaging module gets the data from FPGA through AHB bus interface. We have implemented a control signal in FPGA which processor polls continuously before it starts reading the data.

3.1 Averaging

Window averaging is implemented by taking average of eight samples (one sample from each of 8 windows) with the help of ring buffer as depicted below. 128 locations in window buffer is called bank, samples are stored starting from bank 0, when the buffer is filled with samples of 8 windows, then the next incoming window is stored in bank 0. Once the entire bank has the valid data averaging process starts. Average of 8 windows is calculated and stored in to temporary buffer and sent to Noise cancellation module for further processing.

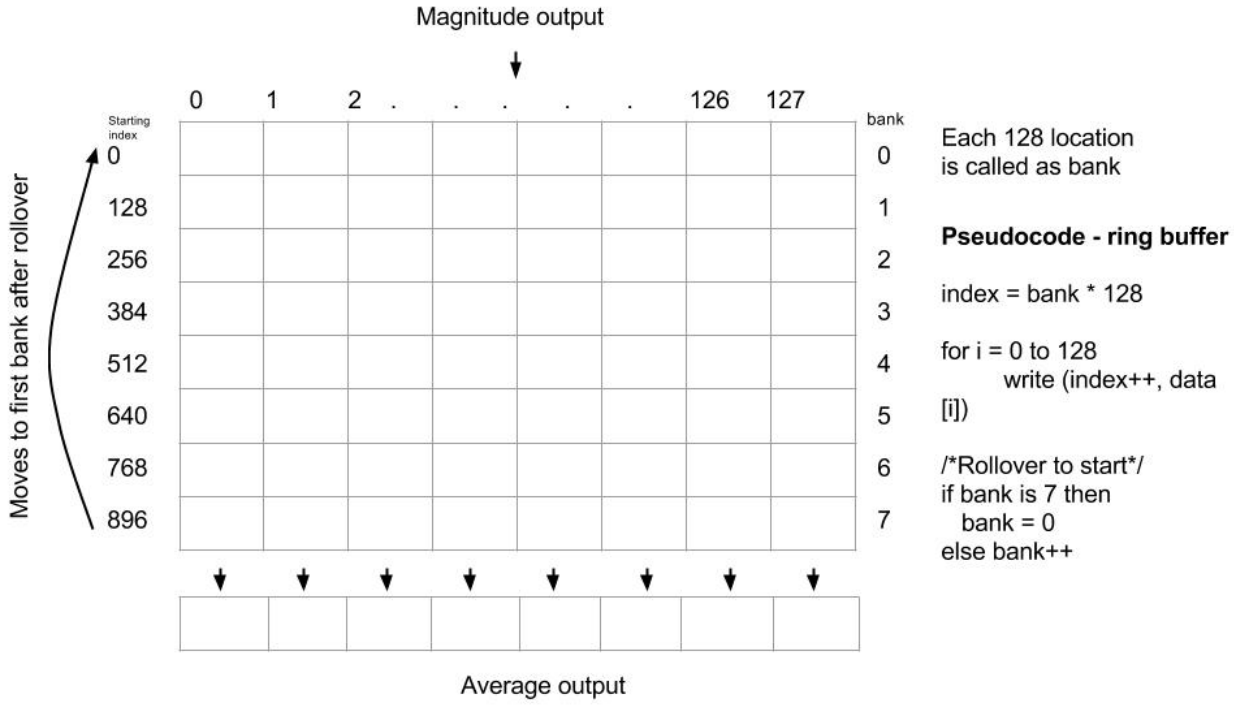


Figure 2 Window buffer structure

3.2 Noise Cancellation

Adaptive noise cancellation technique is used to cancel the ambient noise prevailing in the samples. This technique is quite advanced than just subtracting constant noise value, since the system can adapt to any situation according to noise in the place. Ambient noise is calculated for the first one second to measure the noise in the vicinity which can be subtracted from samples to effectively remove the noise.

4 Optimizations

Without optimizations the latency was close to 110 us excluding *get_audio* function, after applying the below mentioned optimizations latency is reduced drastically to 30us. Most of the optimizations are based on symmetric property of FFT. According to the symmetric property of FFT, if the input sequence is real then the FFT spectrum of that sequence is symmetric in nature. In mathematical form this property is expressed as

$$X[N-k] = X[k]$$ where N implies it is N point FFT.

Followings are the optimizations performed for our design.

1. **DMA** is the key to optimize the execution time where burst of data is transferred from PS to PL instead of single read/write transfer.
2. Moving FFT logic from PS to PL saved around 25 us of execution time.
3. Average module is restricted to perform only 64 computations instead of 128. This technique saves 512 memory read, 64 memory write and 448 additions per each 128 average samples.

5 Experiments

5.1 Area Utilization

Figure 3 shows the resource utilization of our design. We could manage to optimize the resource utilization to 1071 Slice registers and 818 LUTs. Resource utilization with Xilinx FFT IP core 7.1 is very high compared to the FFT 8.0 which helped us to bring the utilization factor very less.

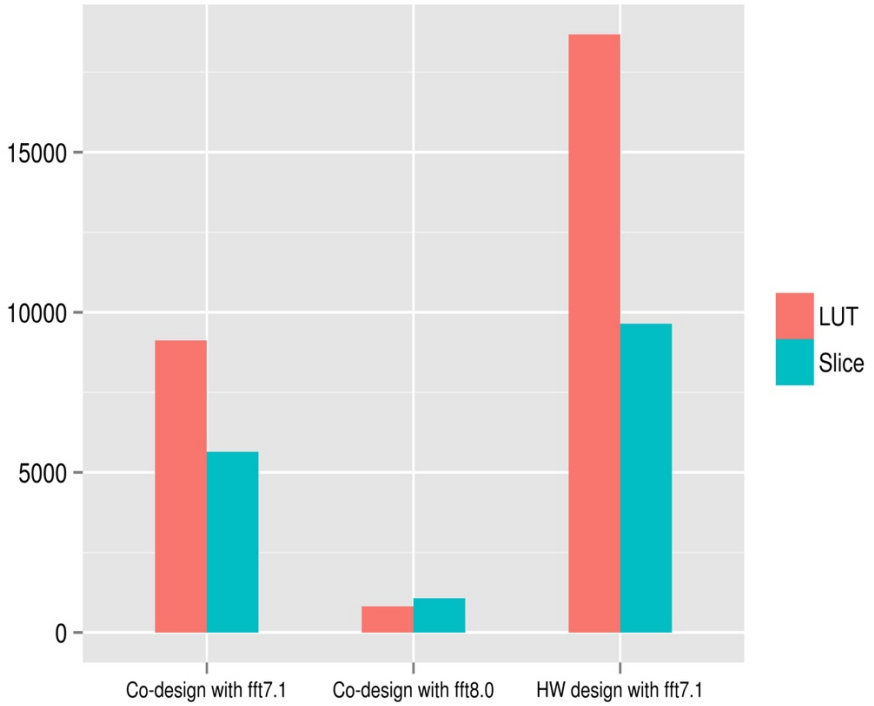
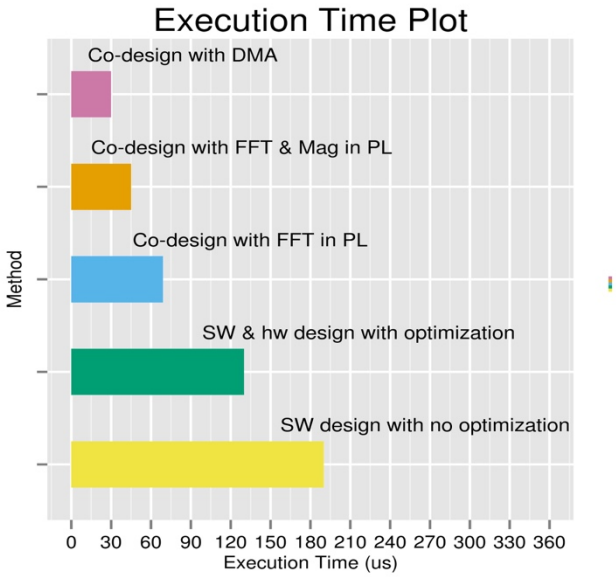


Figure 3 Area Utilization

5.2 Execution time

Figure 4 shows the execution time of various design phase of the system. Out of all the methods software and hardware co-design with DMA gives the best results with software alone design performing the worst. We could able to improve the performance by 30us by moving the magnitude module also to FPGA along with FFT, because the amount of data to be transferred becomes half. By implementing the DMA we could further reduce the time taken by 20us to bring the final execution time to 30us.

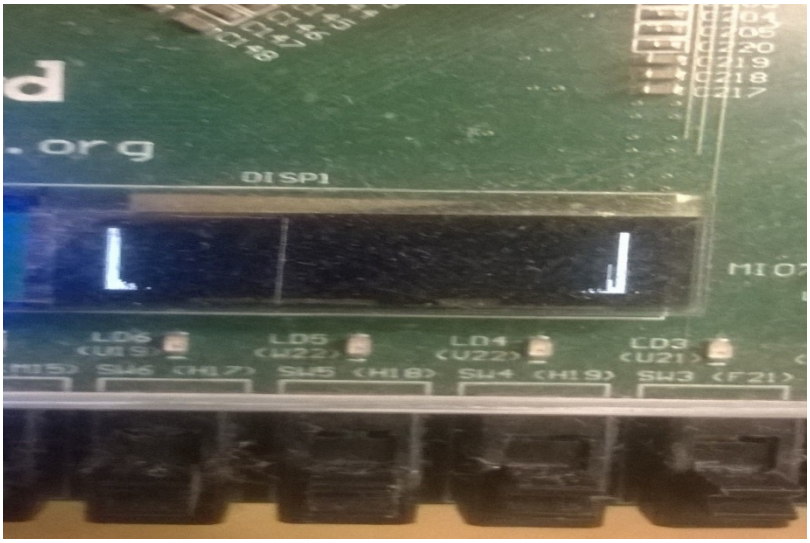


6 Conclusion

Our experiments show that hardware and software co-design is the most efficient way to implement a cost efficient design. It is evident from the results that the co-design implementation is better by 50% when we compare it with software only design. Once the design of the system is made without any flaw co-design takes fullest advantage of the architecture of Hardware and provides better performance.

7 Results

Below figure shows the FFT spectrum output of 20 KHz sine wave



References

1. http://www.fpgadeveloper.com/2014/03/using-the-axi-dma-engine.html
2. http://www.xilinx.com/support/documentation/ip_documentation/ds808_xfft.pdf