IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

HARDWARE SOFTWARE CO-DESIGN USING FPGA AND GNU RADIO ON ZEDBOARD PLATFORM

[1]Pranav M. Tengse, [2]Dr. Nitesh B. Guinde,

[1]Student, [2]Associate Professor

[1]Electronics and Telecommunication Engineering,

[1]Goa College of Engineering, Goa, India

Abstract: *This paper proposes the methodology for implementing an IP core for the FPGA and custom GNU radio block for a simple flowgraph in GNU radio to be run on Soc. A simple IP core such as multiplier is implemented in Hardware Description Languages (HDL), and driven as GNU Radio Block. The focus of this paper is on testing the implemented design on Zedboard and GNU radio in combination with Vivado 2016.4. Firstly in the beginning of the paper we will have a general introduction to a common method to program any FPGA device. Further, it has description and specification of the Zedboard and FPGA used. A brief description about the softwares used. An overview of the custom IP design and AXI peripheral bus has been discussed in the paper. This paper also gives a brief idea about building own project and creating a binary file and creation bootable SD card for Zedboard with the custom IP in it. Finally on the Software side, about GNU Radio and creating custom Blocks in GUN Radio Companion(GRC) is discussed in the paper. Analog Devices supports this implementation methodology. This paper is concluded with possible modification in the proposed methodology.*

***Index Terms* - Zynq-7000 SoC, Zedboard, Vivado Design Suite, IP core, GNU Radio Companion, OOT module**

I. INTRODUCTION

Field Programmable Gate Array(FPGA) is an empowering technology for application oriented systems providing a means for rapid prototyping and evaluation, as well as algorithm acceleration. Many FPGA vendors have recently started experimenting with embedded processors in their devices, like Xilinx with ARM CortexA cores, together with Programmable logic cells. These are known as Programmable System on Chips (PSoC). Their ARM cores(embedded in the processing system or PS) communicates with the Programmable Logic Cells (PL) using ARM standard AXI buses. The Hardware used in this project is Zedboard along with the pre-installed GUN Radio which constitutes the Software side.

In the Hardware part an IP core is designed to perform a particular function. This design can be integrated into a system design. An IP core is designed in HDL(Hardware description language) which is integrated into a system design. The easiest way of integration is by generating a bit stream and port it to FPGA directly. But the problem is that the processor cannot interact with the IP so it cannot be used for accelerating purpose. But if we create a bootable SD Card that allows the user designed core to interact with the processor. Here GNU radio Companion is used to create a Out-Of-Tree (OOT) module which runs on the processor but interacts with the FPGA IP core to speed up the flow of the Flowgraph.

This paper discusses about the design of the IP and the method of integrating it with the GNU radio Companion(GRC) flowgraph. Any system has multiple processes to complete a function. Each process takes a certain number of clock cycles to complete the individual task. Some processes takes a lot of clock cycle which deteriorates the system performance. To overcome this problem Application Specific Integrated circuits(ASICs) can be used. But ASICs are not reprogrammable. Thus, FPGAs which are reprogrammable and also as an improvement to the system performance can be used. Zedboard is a platform designed with Zynq-7000 SoC, using which the proposed design is made.

II. ZEBOARD

The Zedboard is an evaluation and development board based on the Xilinx Zynq7000 processing platform. Interfacing a dual cortex-A9 processing system(PS)with 85,000 series-7 programmable logic(PL) cells, this Zynq-7000 can be marketed for a broader use in many applications. The Xilinx Vivado suit offers the designer a variety of environment to work on the Zedboard that includes flexible design,support for designing and testing of IP blocks which can be used and reused for various design approaches. Thus, Zynq-7000 AP SoC can be targeted for broad use in many applications. The Zedboard's robust mix of non-board peripherals and expansion capability make it an ideal platform. This board contains every-thing necessary to create a Linux, Android, Windows or other OS/RTOS based Design. The Zynq SoC chip consists of two major sections.

•PS:- Processing System

—Dual ARM Cortex-A9 processor, 866MHz to 1GHz frequency

—Multiple peripherals

—Hard silicon core

—Dedicated DDR memory controller

•PL:- Programmable logic

—Logic cells 28k-48k

—AD converters, two 12 bits

—Provides the user the ability to develop their own custom logic or IP which can be software running on processor core.

The essential part of the Zedboard is that the PS and the PL are both linked by a series of interfaces which follow AMBA AXI4 interconnect standard. These interfaces enables the designer to implement custom logic blocks according to their requirements in the PL which can then be connected to the PS and also extend the range of peripherals that are on the AXI bus are easily available and are visible on to the processor's memory map through the use of software.

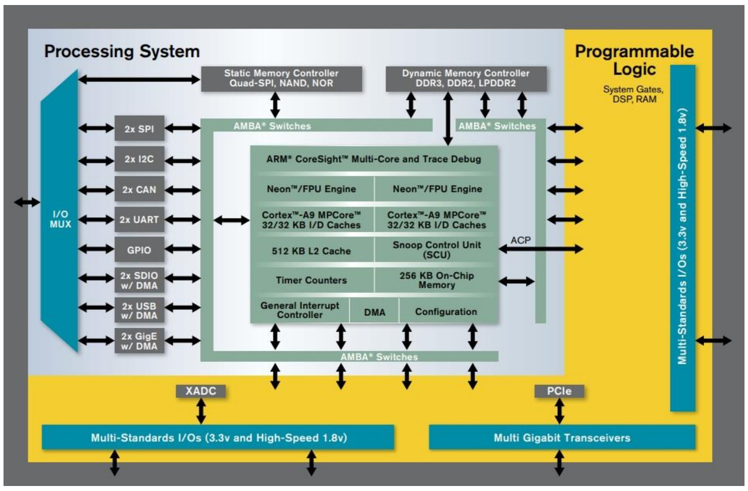
Processing System
Programmable Logic
System Gates, DSP, RAM
Static Memory Controller Quad-SPI, NAND, NOR
Dynamic Memory Controller DDR3, DDR2, LPDDR2
AMBA® Switches
AMBA® Switches
I/O MUX
2x SPI
2x I2C
2x CAN
2x UART
GPIO
2x SDIO w/ DMA
2x USB w/ DMA
2x GigE w/ DMA
ARM® CoreSight™ Multi-Core and Trace Debug
Neon™/FPU Engine
Neon™/FPU Engine
Cortex™-A9 MPCore™ 32/32 KB I/D Caches
Cortex™-A9 MPCore™ 32/32 KB I/D Caches
512 KB L2 Cache
Snoop Control Unit (SCU)
ACP
Timer Counters
256 KB On-Chip Memory
General Interrupt Controller
DMA
Configuration
AMBA® Switches
XADC
PCIe
Multi-Standards I/Os (3.3v and High-Speed 1.8v)
Multi-Standards I/Os (3.3v and High-Speed 1.8v)
Multi Gigabit Transceivers

***Figure 1**: zynq-7000 soc internal architecture*

III. VIVADO AND IP CORE DESIGN

The vivado design Suite[4] is a software, which is designed to improve the productivity. It increases the overall productivity for designing, integrating and implementing with he zynq 7000 based devices. This software replaces the existing Xilinx ISE design, with place and route tools. These tools optimize timing, congestion, total wire length utilization and power.

The Proposed design uses design suite to design the IP core. As mentioned earlier, this IP core is a software core designed in HDL, and tested for its functionality. Ant VHDL or Verilog core can be designed according to the functionality required by the designer. This core can be simulated and tested using testbench. The core is then connected to the AXI peripheral bus, using the procedure explained in [5].

Xilinx's AXI peripheral bus is a pare of ARM AMBA bus[6]. There are 3 types of AXI4 interfaces as mentioned below.

- AXI4:- it provides high performance memory mapped requirement.
- AXI4-lite :- it provides low throughput memory mapped communication.
- AXI4-stream :- It provides high speed streaming data.

In the design of AXI peripheral IP core. AXI4 Lite slave is used. It consists of 32 bit data width, 64bytes memory and 4 registers. As mentioned in [5], A simplest of IP core like multiplier, or an IP core that performs computationally expensive task like FIR Filter or FFT can be added to the AXI bus. According to the user designed IP core, the peripheral bus designs need to be modified, so that data control signals propagates as desired. The IP core designed in this method needs to be packed and added to the repository as mentioned in [5] The used designed IP present in the user repository can then be added to the system design.

3.1 Building the project

ADI(Analog Devices Inc) provides the source files needed to create and build the design. Thus the burden of modifying and building these projects is on the designer. The building process depends on a certain software and tools. In this proposed method vivado 2016.4 and SDK support was chosen. The procedure suggested in [7] successfully builds a desired project for fmcomms2 and zedboard combination. The project build is shown in Figure 2. The used designed IP core can be added to this project as an AXI peripheral.

On generating the system design and adding user designed IP core to it, the design was synthesized and implemented. The bitstream was generated and an ".hdf' file was generated. This file consists of the system design specifications and provides the physical address of the use designed IP. This file can be read in Xilinx's SDK. Xilinx Software Development Kit(XSDK) is a software which is designed create embedded applications on any of the Xilinx's microprocessors like Zynq Ultrascale, Zynq 7000 SoCs etc. It provides homogeneous and heterogeneous multiprocessor design, debug, and performance analysis it provides many advantages out of which many mentioned below.

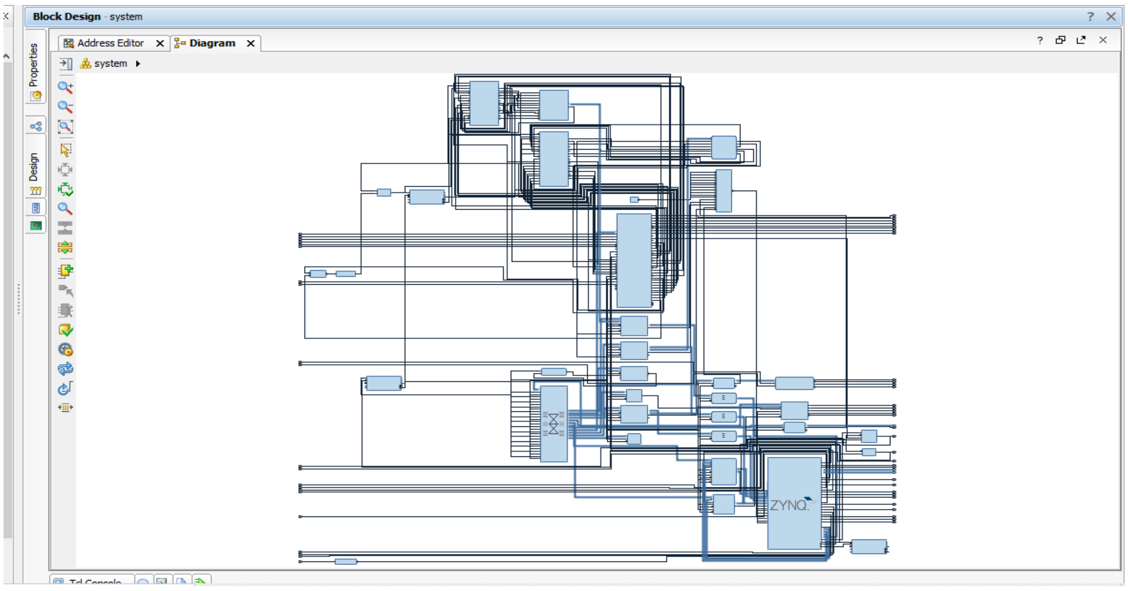

Figure 2: *system project build for zedboard*

On generating the system design and adding user designed IP core to it, the design was synthesised and implemented. The bitstream was generated and an ".hdf' file was generated. This file consists of the system design specifications and provides the physical address of the use designed IP. This file can be read in Xilinx's SDK. Xilinx Software Development Kit(XSDK) is a software which is designed create embedded applications on any of the Xilinx's microprocessors like Zynq Ultrascale, Zynq 7000 SoCs etc. It provides homogeneous and heterogeneous multiprocessor design, debug, and performance analysis it provides many advantages out of which many mentioned below.

- Included with vivado deign suite.
- It provides complete integrated design environment that directly interface with the hardware design environment.
- It delivers a complete software design and debug flow supported.
- The SDK provides an editor, compiler, build tool, flash memory management, and JTAG debug integration for successful execution of software part of the IP design.

IV. BOOTABLE SD CARD

A lot of research has already been done to make a bootable SSD card, which supports Xilinx FPGA devices. But [9] provides a script to make a "boot.bin" file that will support the FMCOMMS2 and zedboard combination. The process to build the kernel, the device tree, the Zynq Boot image is explained in details. It provides a script which takes ".hdf" file and the "uboot.elf" file and generated the "boot.bin" file. This file can now directly be copied to the SD Card to make it bootable. The process required to generate the "uboot.elf" is described in [9] A "boot.bin" file was implemented successfully for the above system design.

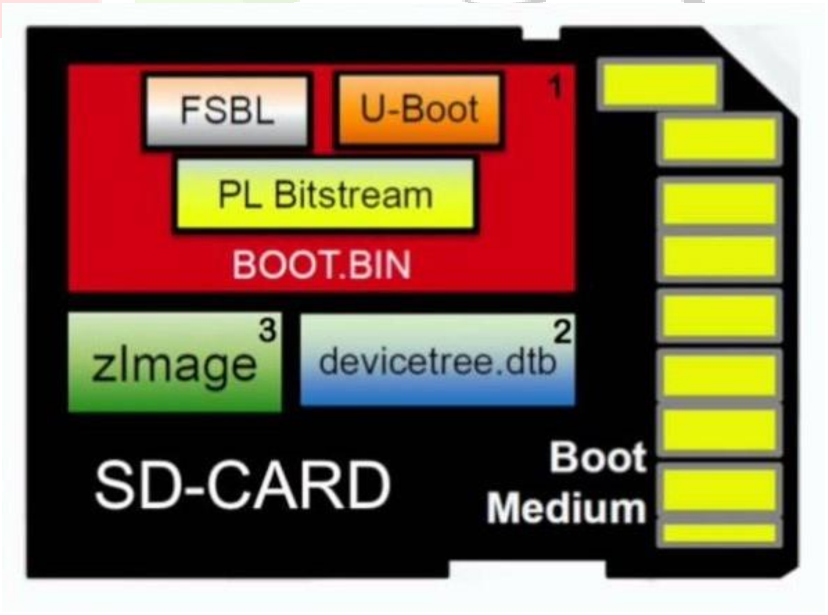

Figure 3: basic files included for the operation of embedded linux on zedboard

V. GNU Radio Conpanion and OOT module

Software defined radios (SDRs) have changed the paradigm of slowly designing custom radios, instead allowing designers to quickly iterate designs with a large range of functionality. GNU Radio Companion(GRC) is free software that can be used in Linux based OS (Ubuntu) for simulating communication systems and for designing Software Defined Radios.

In the proposed methodology OOT(out of tree) module is created in GRC which uses the custom IP core in FPGA to do the task. A custom GUN radio block is an OOT (out of the tree) module. An out-of-tree module is a GNU Radio component that does not live within the GNU Radio source tree. Typically, if you want to extend GNU Radio with your own functions and blocks, such a module is what you create . This allows you to maintain the code yourself and have additional functionality alongside the main code.

To build the block the steps are described in [10].Once the module is installed, the block in the module can be used in a flowgraph. And it will work as a performance enhancer in the flowgraph.

The block uses mmap() interface[9] to allow access to kernel buffer and the address space reserved to AXI Lite slave bus. The block code must include the base addrress of the AXI peripheral IP obtained from the “hdf” file. It must also consists of input signals to be sent to the IP and output is stored on a bus register which is read for displaying.

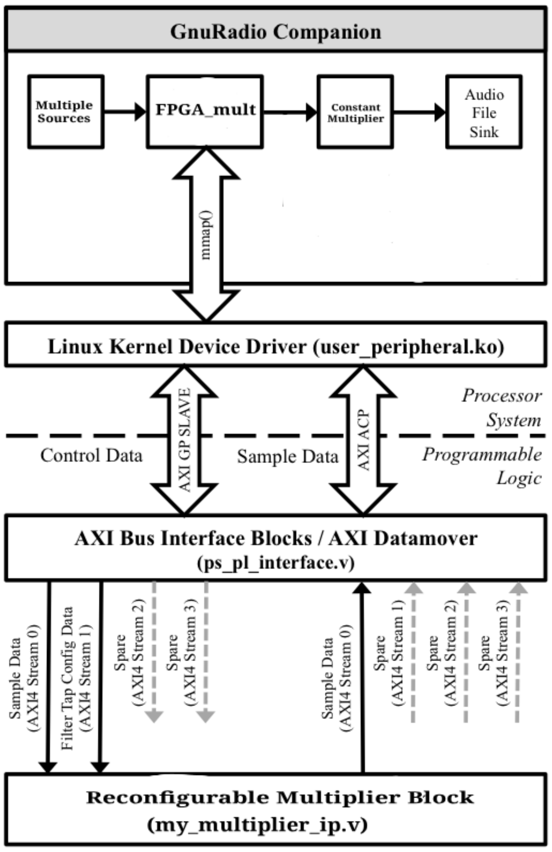

Figure 4: *block diagram for proposed design*

VI. Conclusion

A methodology for implementing a user designed IP core that was used the user designed GRC module to speed-up the processes of the GRC flowgraph has been described in this paper. Although the word is confined to Zedboard and GRC , implementation for the other combinations of Zynq platforms is possible. Similarly other software tools can be used.

There are many research groups that are working on this area and published their contribution in the study of FPGAs. But finding a methodology that describes the entire process, for a specific platform was a challenge

Future scope of the proposed work would be to implement a computationally expensive IP core, that can be integrated in the GRC. Also this work can be used for other applications like video processing, image processing, and signal processing. Such applications include transform computations which increase system latency. The performance speed, efficiency and throughput of such applications can be improved

VII. Acknowledgment

The authors would like to thank the members of Xilinx and ADI. We also thank the institution for providing necessary equipment involved in the study

References

[1] M. A. Kadi, P. Rudolph, D. Gohringer and M. Hubner, "Dynamic and partial reconfiguration of Zynq 7000 under Linux," 2013 International Conference on Reconfigurable Computing and FPGAs (ReConFig), Cancun, 2013, pp. 1-5, doi: 10.1109/ReConFig.2013.6732279
[2] Y. Han and E. Oruklu, "Real-time traffic sign recognition based on Zynq FPGA and ARM SoCs," IEEE International Conference on Electro/Information Technology, Milwaukee, WI, 2014, pp. 373-376, doi: 10.1109/EIT.2014.6871793.
[3] Zedboard Getting Started Guide Version 7.0.2017, Avenet, Inc. AVNET
[4] https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_2/ug910-vivado-getting-started.pdf
[5] https://www.fpgadeveloper.com/2014/08/creating-a-custom-ip-block-in-vivado.html/
[6]https://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/latest/ug1037-vivado-axi-reference-guide.pdf
[7] https://wiki.analog.com/resources/fpga/docs/build
[8] https://wiki.analog.com/resources/eval/user-guides/ad-fmcomms2-ebz/software/linux/zynq_2014r2
[9] https://man7.org/linux/man-pages/man2/mmap.2.html
[10]https://wiki.gnuradio.org/index.php/OutOfTreeModules