Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 13 Issue 07, July-2024

# Video input Through Test Pattern Generator IP Core on Zybo Board (Zynq 7000)

Karthik Poduval

***Abstract*—This paper demonstrates an example video input device using TPG IP on a Xilinx ZYNQ7000 series SoC/**

***Index Terms*—FPGA, Video Capture**

## I. INTRODUCTION

The goal of this project is to build and SoC using the Zynq 7000 series processor using the programmable logic block to implement a video Test Pattern Generator and other necessary blocks to generate video and transmit into the ZYNQ. Before describing the hardware and software architecture for this project, it makes sense to first describe the zynq architecture first in the next few sections covering the aspects that matter the most to this project.

## II. ZYNQ ARCHITECTURE

The Zynq chip used in this project is the Zynq 7000 series SoC. All the Zynq variants under this family share the same architecture with the exceptions that variants may have different amounts of programmable logic. Figure 1 depicts a

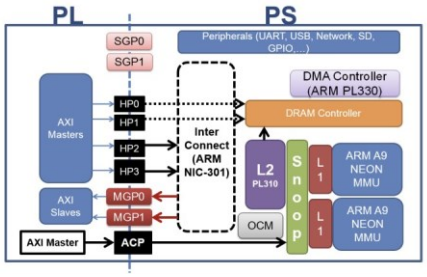


Fig. 1. ZYNQ7000 Architecture

typical Zynq architecture. The Processing System (PS) consists of a dual ARM Cortex A9 processors with dedicated L1 caches and a shared L2 cache. The two L1 cache units are coherent through a snoop control unit. Access to the snoop control unit (SCU) is also made available to external hardware through the ACP such that AXI masters on the PL1 can implement cache coherent access to memory blocks.

### A. Programmable Logic

On the PL side of things, we have general purpose configurable logic blocks completely managed by the Vivado software package which translate from IP RTL to implementation on the PL system. The IOB (Input Output Blocks) allow for external word interfacing though pins present on the pin package. The PL also contains DSP blocks and block RAM for specialized implementations that can make use of such accelerators.

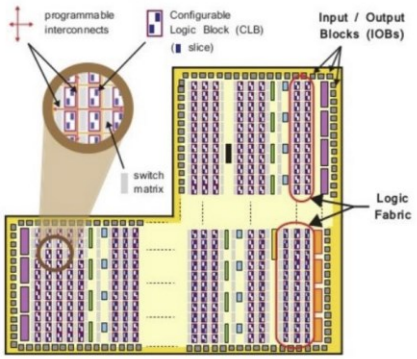


Fig. 2. The programmable logic fabric

### B. PS PL Interconnect

To connect the PL and PS together, the ZYNQ uses the AXI interface/ interconnect as shown in Figure 3. In addition, there are 9 AXI interfaces to and from the PL system. These AXI interfaces fall under the following categories.

- General Purpose (GP) AXI Interface: These are general purpose as the name suggests and allows for general communication between the PS and PL. We have Master interfaces where PS is master and PL is slave and Slave interfaces where PL is master and PS is slave. The GP interfaces have no buffering and are suitable for low speed data transfers and memory mapped IO operations.
- Accelerator Coherency Port: The ACP interface is a 64 bit AXI interface which allows for a master on PL to make transfers to PS memory elements that are cache coherent.
- High Performance (HP) AXI Interface: These interfaces are buffered (with FIFOs) and suited for bursts transfers between master and slave. All the HP interfaces are slave interfaces w.r.t PS (PL is the master). VDMA (video DMA, which we will cover in a later section, gains access to the memory controller on PS through this interface).

Zybo [1] is a development/prototyping board designed and manufactured by Diligent Inc. Zybo contains the Z-7010 variant of the Zynq 700 series family. Following are the key features of the Zybo board.

### C. Zybo Architecture

Zybo is a development/prototyping board designed and manufactured by Diligent Inc. Zybo contains the Z-7010 variant of the Zynq 700 series family. Following are the key features of the Zybo board.

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 13 Issue 07, July-2024

- Xilinx Zynq-7000 XZ7Z010-1CLG400C

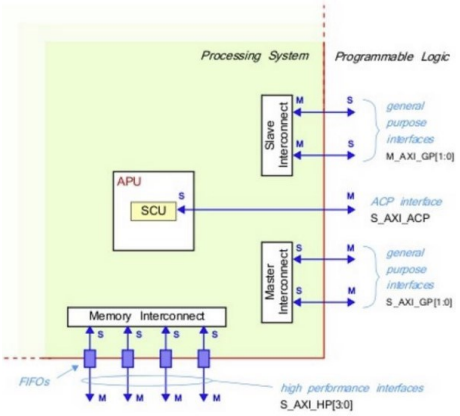


Fig. 3. PS PL Interface/Interconnect

- 28000 logic cells
- 240 KB Block RAM
- 80 DSP slices
- On-chip dual channel, 12-bit, 1 MSPS analog-to-digital converter XADC)
- 650 MHz dual-core Cortex™-A9 processor
- On-board JTAG programming and UART to USB converter
- DDR3 memory controller with 8 DMA channels
- 512 MB x32 DDR3 w/ 1050Mbps bandwidth
- 128 Mb Serial Flash w/ QSPI interface
- microSD slot (supports Linux file system)
- High-bandwith peripheral controllers: 1G Ethernet, USB 2.0, SDIO
- Low-bandwidth peripheral controller: SPI, UART, I2C
- Dual-role (Source/Sink) HDMI port
- 16-bits per pixel VGA output port
- Trimode (1Gbit/100Mbit/10Mbit) Ethernet PHY
- OTG USB 2.0 PHY (supports host and device)
- External EEPROM (programmed with 48-bit globally unique EUI-48/64™ compatible identifier)
- Audio codec with headphone out, microphone and line in jacks
- GPIO: 6 pushbuttons, 4 slide switches, 5 LEDs
- Six Pmod ports (1 processor-dedicated, 1 dual analog/digital)

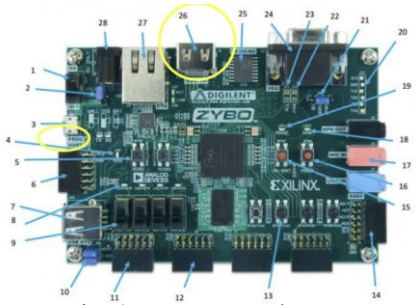


Fig. 4. ZYBO Board

| Callout | Component Description |
|---|---|
| 1 | Power Switch |
| 2 | Power Select Jumper and battery header |
| 3 | Shared UART/JTAG USB port |
| 4 | MIO LED |
| 5 | MIO Pushbuttons (2) |
| 6 | MIO Pmod |
| 7 | USB OTG Connectors |
| 8 | Logic LEDs (4) |
| 9 | Logic Slide switches (4) |
| 10 | USB OTG Host/Device Select Jumpers |
| 11 | Standard Pmod |
| 12 | High-speed Pmods (3) |
| 13 | Logic Pushbuttons (4) |
| 14 | XADC Pmod |
| 15 | Processor Reset Pushbutton |
| 16 | Logic configuration reset Pushbutton |
| 17 | Audio Codec Connectors |
| 18 | Logic Configuration Done LED |
| 19 | Board Power Good LED |
| 20 | JTAG Port for optional external cable |
| 21 | Programming Mode Jumper |
| 22 | Independent JTAG Mode Enable Jumper |
| 23 | PLL Bypass Jumper |
| 24 | VGA connector |
| 25 | microSD connector (Reverse side) |
| 26 | HDMI Sink/Source Connector |
| 27 | Ethernet RJ45 Connector |
| 28 | Power Jack |

Fig. 5. ZYBO Pins (Highliged ones used in this project)

## III. SYSTEM ARCHITECTURE

Now that all the key elements of the Zynq and the Zybo board have been introduced, it's now time to describe the project architecture.

### A. Top Level Setup

At a very high level we have a PC which is used to program the Zybo via JTAG and also communicate with the test software via UART. An HDMI display is connected to the Zybo for display.


Fig. 6. Top Level Setup

The Zybo based system is used as starting point which displays a software generated test pattern on to the HDMI interface. Then I add a Test Pattern Generator(TPG) on top of the Zybo base system design. The test pattern IP core transmits the generated video data to the DDR memory though a VDMA (Video DMA) IP block. Once this happens the software generated test pattern is overwritten by the video data DMA'd by the TPG core.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 13 Issue 07, July-2024

B. Detailed Design Architecture

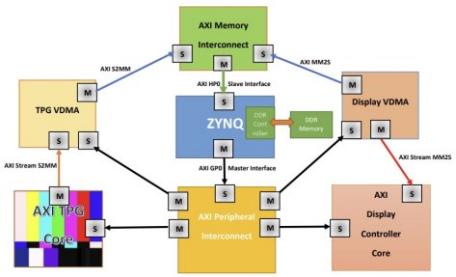


Fig. 7. Detailed Design

In this section lets describe the detailed design. Figure 6 outlines the detailed design architecture. The Zynq processing system is added along with one AXI GP0 master interface and a AXI HP0 slave interface. The AXI GP0 master interface is connected to an AXI Peripheral Interconnect. It uses the AXI Lite protocol to talk to all the different IP blocks in the design (display controller, TPG, display VDMA, TPG VDMA). AXI Lite is an AXI4 protocol used primarily for programming the different IP blocks through memory mapped IO interface. The Zynq is also a slave to the AXI HP0 interface. On the other end of the AXI HP0 interface is an AXI Memory interconnect. The Display VDMA and the TPG VDMA's are masters as the have direct access to the DDR memory though the HP0 interfaces. The AXI TPG core interfaces to the AXI VDMA through the AXI Stream protocol which is high speed and allows for burst transfers. The AXI VDMA is a write master and the Display VDMA is a read master w.r.t DDR memory. Once the DMA's have been initialized and provided with memory addresses, they start moving data between hardware IP (TPG, Display Controller) and memory. The DMA's can be configured to generate interrupt on completion as well as indicating errors, however this design doesn't include the GIC (Generic Interrupt Controller) so we do not have this functionality.

## IV. SOFTWARE ARCHITECTURE

The software for this project starts with the Zybo base system software and adds the software to control the TPG and TPG VDMA on top of it. The software flow is shown in Figure 8 and desing files can be found in [2].

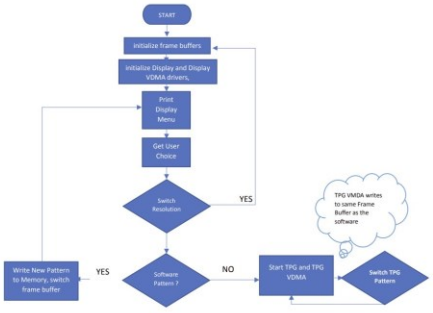


Fig. 8. Software Flow

## REFERENCES

[1] "Zybo." [Online]. Available: https://tinyurl.com/zjr9uht
[2] "Designfiles." [Online]. Available: https://github.com/karthikpoduval/ ZYBO-TPG