دانشگاه صنعتی شریف

دانشکده مهندسی برق

طراحی سیستم‌های مبتنی بر ASIC/FPGA

گزارش بخش PL فاز 1 پروژه

استاد: دکتر شعبانی

تهیه‌کنندگان:

امیررضا ایمانی 99101279

محمدپارسا عابدی 99106385

محمدرضا حاجی‌بابایی 99101416

Introduction

This report provides a comprehensive guide on preparing the Programmable Logic (PL) part of a project on the Zynq7010 platform. The project involves generating various types of waveforms in the Processing System (PS) side, transferring them to the PL via Direct Memory Access (DMA), performing Fast Fourier Transform (FFT) on the signals, and displaying both the original and transformed signals on a monitor using HDMI.

Project Overview

The project workflow includes the following key steps:

1. **Waveform Generation in PS**: Different types of waveforms are generated.
2. **DMA Transfer to PL**: Generated signals are sent to the PL for processing.
3. **FFT Calculation in PL**: FFT IP Core computes the Fourier transform of the signals.
4. **DMA Transfer to PS**: Processed signals are sent back to the PS.
5. **VDMA and HDMI Display**: Both original and transformed signals are displayed on a monitor.

Block Diagram

The project's block diagram (see Figure 1) illustrates the flow from signal generation to display.

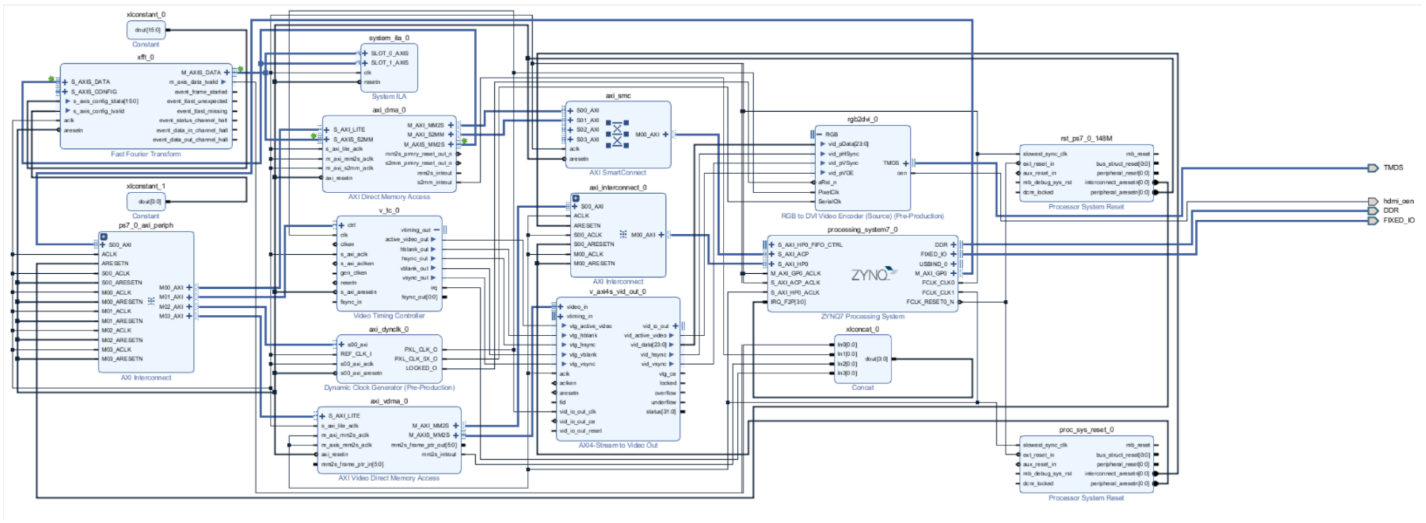


Figure 1: Block Diagram of the Project

Detailed Steps and Configuration

Step 1: Waveform Generation in PS

The PS is responsible for generating different types of waveforms, such as sine, triangular, and square waves. This can be done using a signal generator function in the PS. The generated signals are then prepared for transfer to the PL.

Step 2: DMA Transfer to PL

To transfer the generated signals from the PS to the PL, we configured the DMA as follows:

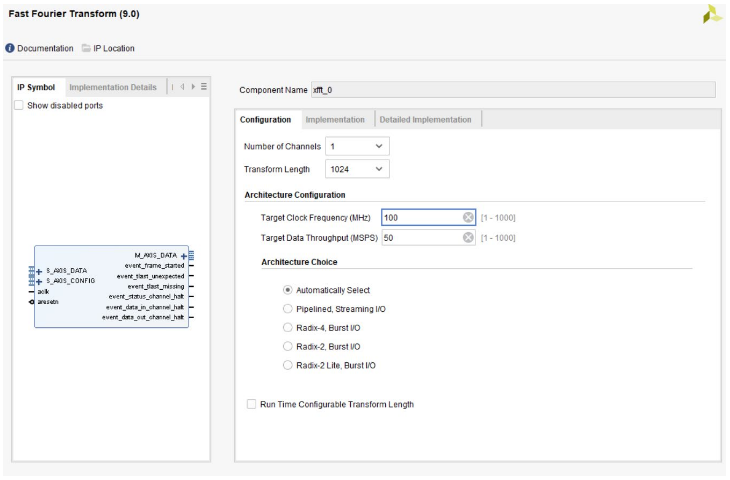
- **HP (High Performance) Ports**: Use the HP ports on the Zynq PS to connect to the AXI DMA in the PL.
- **DMA Configuration**: Set up the DMA to handle streaming data from the PS to the PL.

Step 3: FFT Calculation in PL

In the PL, we used the FFT IP Core to perform the Fourier transform on the received signals. We configured the FFT IP Core with the following settings:

- **Number of Channels**: 1
- **Transform Length**: 1024
- **Data Format**: Fixed Point
- **Input Data Width**: 8 bits
- **Output Ordering**: Natural Order

FFT IP Core Configuration



Fast Fourier Transform (9.0)

Show disabled ports

Component Name: xfft_0

Number of Channels: 1

Transform Length: 1024

Architecture Configuration

Target Clock Frequency (MHz): 100 [1 - 1000]

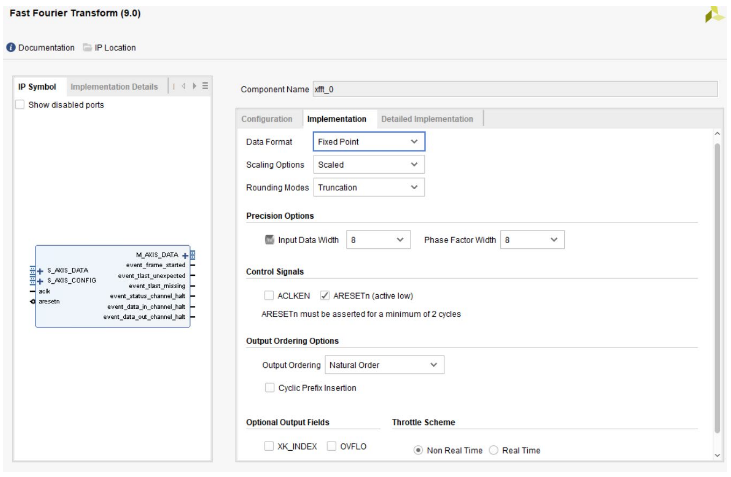
Target Data Throughput (MSPS): 50 [1 - 1000]

Architecture Choice

[✓ Automatically Select]
Pipelined, Streaming I/O
Radix-4, Burst I/O
Radix-2, Burst I/O
Radix-2 Lite, Burst I/O

Run Time Configurable Transform Length

Figure 2: FFT IP Core Configuration (Part 1)



Component Name: xfft_0

Data Format: Fixed Point
Scaling Options: Scaled
Rounding Modes: Truncation

Precision Options

Input Data Width: 8 Phase Factor Width: 8

Control Signals

☐ ACLKEN ✓ ARESETn (active low)

ARESETn must be asserted for a minimum of 2 cycles

Output Ordering Options

Output Ordering: Natural Order

☐ Cyclic Prefix Insertion

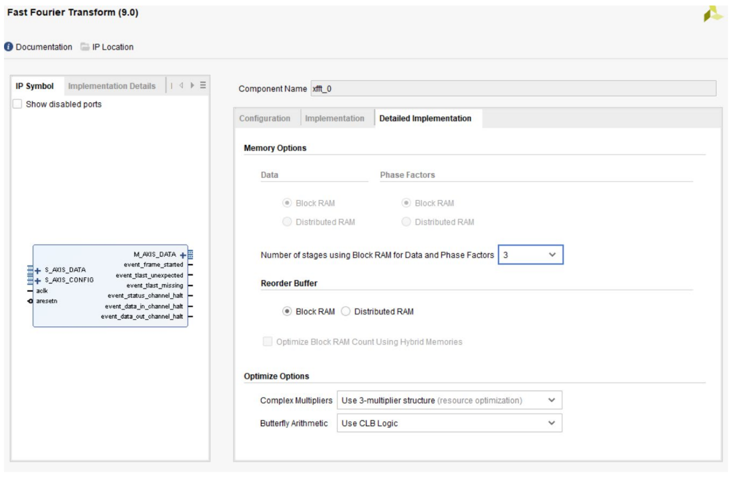
Optional Output Fields

☐ XK_INDEX ☐ OVFLO

Throttle Scheme

[✓ Non Real Time] Real Time

Figure 3: FFT IP Core Configuration (Part 2)



Component Name: xfft_0

Memory Options

Data: [✓ Block RAM] Distributed RAM

Phase Factors: [✓ Block RAM] Distributed RAM

Number of stages using Block RAM for Data and Phase Factors: 3

Reorder Buffer

[✓ Block RAM] Distributed RAM

☐ Optimize Block RAM Count Using Hybrid Memories

Optimize Options

Complex Multipliers: Use 3-multiplier structure (resource optimization)
Butterfly Arithmetic: Use CLB Logic

Figure 4: FFT IP Core Configuration (Part 3)

Step 4: Add and Configure Dynamic Clock Generator

1. We added the Dynamic Clock Generator IP to the block design.
2. We configured it to generate the necessary clocks for the system, including the clock for the HDMI output.

Step 5: DMA Transfer to PS

After the FFT computation, the transformed data is sent back to the PS through another DMA. The configuration is similar to the previous DMA setup but in the reverse direction.

Step 6: Add and Configure RGB to DVI Video Encoder

1. We added the RGB to DVI Video Encoder IP to the block design.
2. We connected it to the VDMA output for converting the video data to a format suitable for HDMI display.

Step 7: VDMA and HDMI Display

The PS uses the Video Direct Memory Access (VDMA) controller to send the original and FFT signals to the HDMI controller for display on a monitor.

IPs Used

- **AXI DMA**: For transferring data between PS and PL.
- **FFT IP Core**: For computing the Fourier transform of the signals.
- **AXI VDMA**: For handling video data transfer to HDMI.
- **Dynamic Clock Generator**: For generating necessary system clocks.
- **RGB to DVI Video Encoder**: For converting video data to HDMI format.
- **HDMI Controller**: For displaying data on a monitor.

Block Design

The detailed block design includes the configuration and interconnection of various IPs used in the PL.

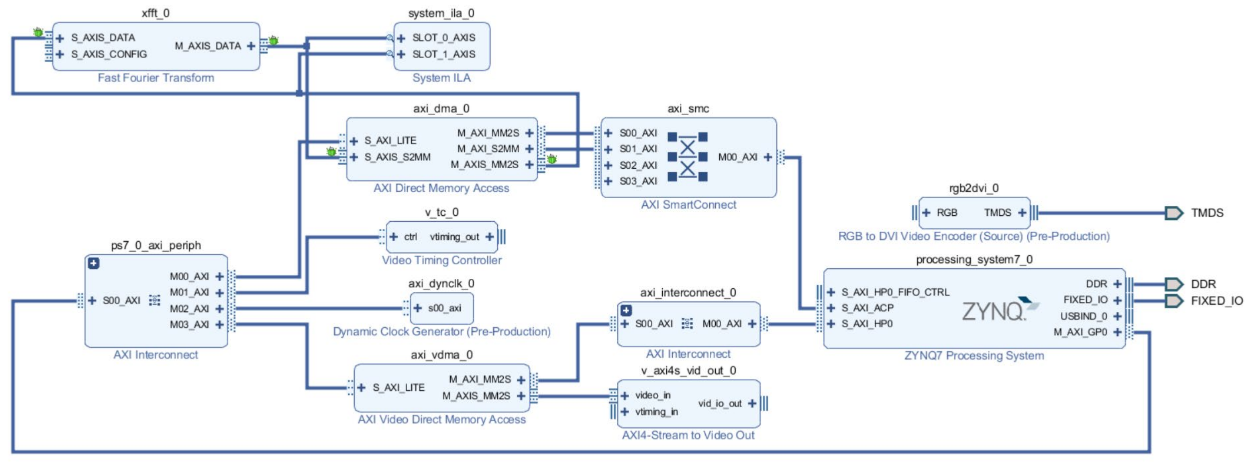


Figure 4: Abstract Block Design

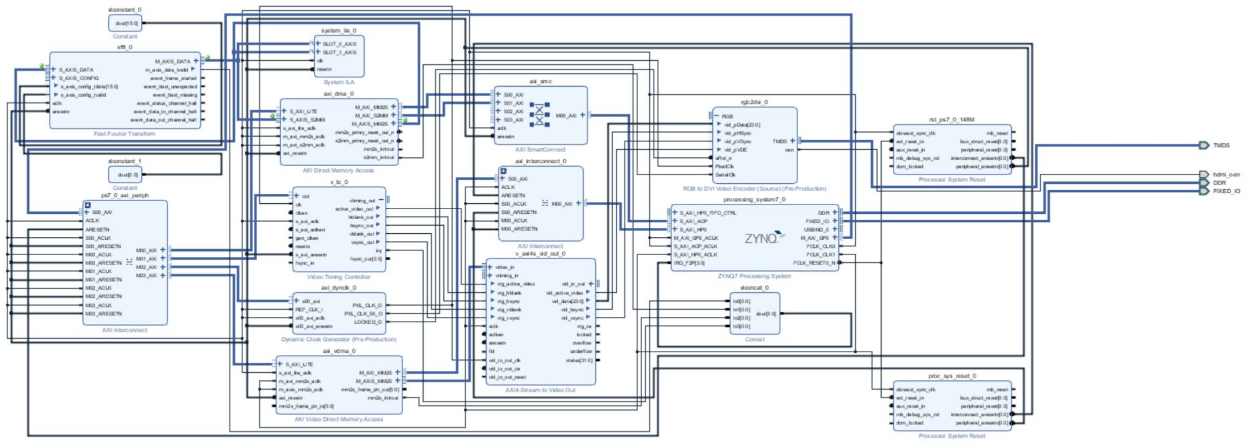


Figure 5: Complete Block Design

Implementation Steps

1. Configure PS

- We configured the PS block to enable necessary peripherals (HP ports for DMA, DDR memory, etc.).
- We connected PS to DDR and other required interfaces.

2. Add IP Repository

- We navigated to **Tools** > **Settings** > **IP** > **Repository**.
- We added the directory containing the custom IPs for Dynamic Clock Generator and RGB to DVI Video Encoder.

3. Add and Configure AXI DMA

- We added the AXI DMA IP to the block design.
- We connected AXI DMA to the HP ports of the PS.
- We configured the AXI DMA for streaming data transfer.

4. Add and Configure FFT IP Core

- We added the FFT IP Core to the block design.
- We configured the FFT IP Core with the settings mentioned earlier.
- We connected the FFT IP Core to the AXI DMA for data input and output.

5. Add and Configure Dynamic Clock Generator

- We added the Dynamic Clock Generator IP to the block design.
- We configured it to generate the necessary clocks for the system, including the clock for the HDMI output.

6. DMA Transfer to PS

- After the FFT computation, we configured DMA to transfer the transformed data back to the PS.

7. Add and Configure RGB to DVI Video Encoder

- We added the RGB to DVI Video Encoder IP to the block design.
- We connected it to the VDMA output for converting the video data to a format suitable for HDMI display

8. Add AXI VDMA and HDMI Controller

- We added the AXI VDMA and HDMI controller IPs.
- We connected AXI VDMA to the PS and configure it for video data transfer.
- We connected HDMI controller to the VDMA output for displaying data on the monitor.

9. Connect and Configure AXI Interconnects

- We used AXI interconnects to connect all IP blocks.
- We ensured all necessary signals (clock, reset, etc.) are properly connected.

10. Generate Bitstream and Export Hardware

- We validated the block design.
- We generated the bitstream for the design.
- We exported the hardware including the bitstream to SDK.

Conclusion

This project demonstrates the integration of PS and PL on the Zynq7010 platform to perform real-time signal processing and display. The configuration of DMA and FFT IP Core in the PL, along with VDMA and HDMI controller in the PS, provides a comprehensive approach to handling digital signal processing tasks efficiently.