DTBDM(Decision-Tree-Based De-noising Method)

Chethan Kumar

Chapter 1

Abstract

The project is aimed at implementation of decision-tree-based impulse noise detector to detect the noisy pixels, and an edge-preserving filter to reconstruct the intensity values of noisy pixels (”ieeexplore.ieee.org/iel5/12/6471716/06122015.pdf”) on Cyclone V soc. We use standard 512x512 test images to validate the design. Input image is sent to FPGA and the processed image is received on PC through USB blaster using Altera JTAG-AVALON bridge IP core. This project is implemented in Qsys tool (Verilog HDL).

Chapter 2

Design

2.1 Original Architecture

Figure 2.1 depicts a single instance of the original MIPS soft processor design along with its various components in the form of a block diagram. The Execution Unit, Instruction Decoder and System Co-Processor mainly make up the Execution Data Path (EDP) which are used to generate the virtual addresses to access the shared memory. The Memory Management Unit (MMU) converts these virtual addresses into physical addresses. The Instruction Cache and Data Cache represent the memories which provide faster access to instructions and data respectively. The Bus Interface Unit BIU) controls the transfer of data between the MIPSfpga soft processor and the shared memory. The Multiplication and Division Unit (MDU) performs the arithmetic operations for the processor.

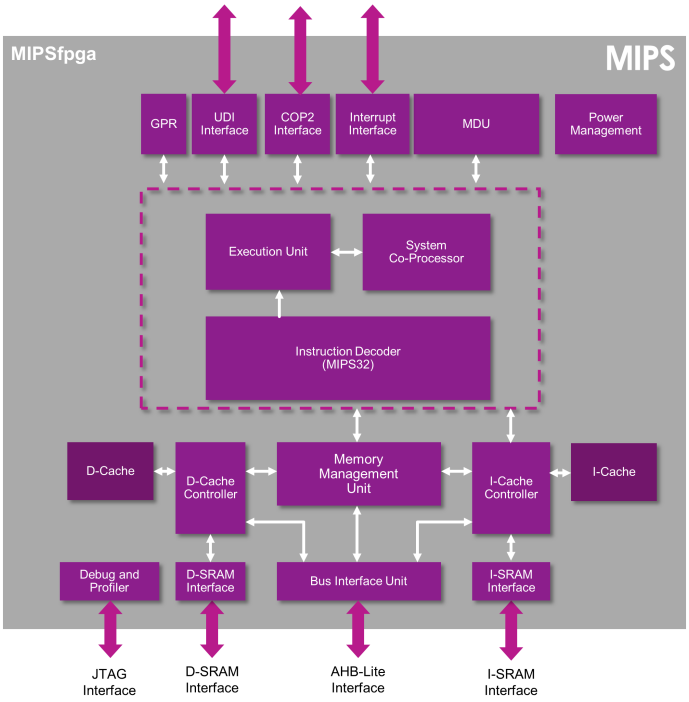

Figure 2.1: Original MIPS Architecture

We will now explain the optimization performed in each of the components of the original design:

2.1.1 Execution Data Path

The EDP was modified to directly generate the addresses of the instructions in the Instruction RAM which was added by us. The instructions from the I-RAM are then passed on to the Instruction Decoder for decoding. The decoded instructions are sent back to the EDP either for arithmetic operations or to access the UDI for transfer of data between different processors.

2.1.2 Memory Management Unit (MMU)

As we are directly generating the address of instructions to be accessed in the I-RAM, we can completely bypass the MMU which used to perform the task of converting virtual addresses to physical addresses previously. It is a redundant component in the new design and is ,therefore, not synthesized by the tool.

2.1.3 Instruction Cache and Data Cache Controllers

Since we wanted to modify the existing architecture into a Distributed shared-memory multiprocessor (DMP), we completely eliminate the I-Cache and D-Cache. However, we cannot completely eliminate their controllers as it generates some critical signals which are required by the Instruction Decoder.

2.1.4 User Defined Instructions

A User Defined Instruction (UDI) unit with different sections for sending and receiving instructions across processors using the user-defined Send and Receive instructions is implemented by us. Each processor has its own scratchpad allowing them to independently access their personal memories and also share data between processors.

2.1.5 Bus Interface Unit (BIU)

The removal of the external shared memory allows us to eliminate the BIU which used to control the transfer of data across the shared memory between processors. However, due to certain critical signals which were being generated by the BIU, we could not completely eliminate it. We have managed to remove all of its redundant functionality but kept certain critical signals like indicating to the EDP when to generate the next address.

2.1.6 Register File

The original design implemented the Register File in the form of registers, each of which was 32-bit wide. In order to save out on the number of registers utilized, we replaced the RF with an overclocked RAM with 2 read ports and one write port. This RAM works at a frequency which is 2x faster than the original circuit frequency. It has been implemented using a Block-RAM with a size of 9 kbits meaning that it can store 256 32-bit instructions.

2.1.7 Multiplication and Division Unit (MDU)

The MDU implemented multiplication using a series of additions in accordance with Booth's Algorithm. We replace this by a DSP block to reduce the number of utilized LUTs. Also support for division was removed from the original implementation. DSP blocks have also been utilized to implement special functions of the processor such as Multiply-Add and signed and unsigned operations.

2.2 Master Pipeline Control (MPC)

The MPC is basically the instruction decoder which gets instructions from the Instruction RAM, decodes them and passes on the decoded instructions to the EDP for execution.

2.3 Optimized Architecture

Based on all the optimizations performed by us, the architecture is as represented in Figure 2.2. We have not been able to completely eliminate certain components due to some critical signals which they were generating and were essential for the operation of the processor on a whole.

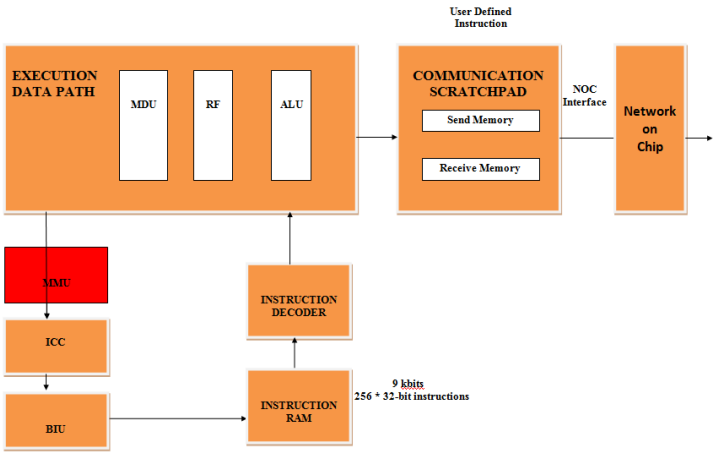

Figure 2.2: Optimized MIPS core

Chapter 3

Methodology

3.1 Simulation

We use Altera ModelSim Starter Edition 10.0 to simulate our design and verify its functionality. We write **tcl** scripts to automate the simulation process. This tcl script is responsible for compiling verilog files, libraries and generating input vectors. Figure 3.1 depicts the test setup of our project.

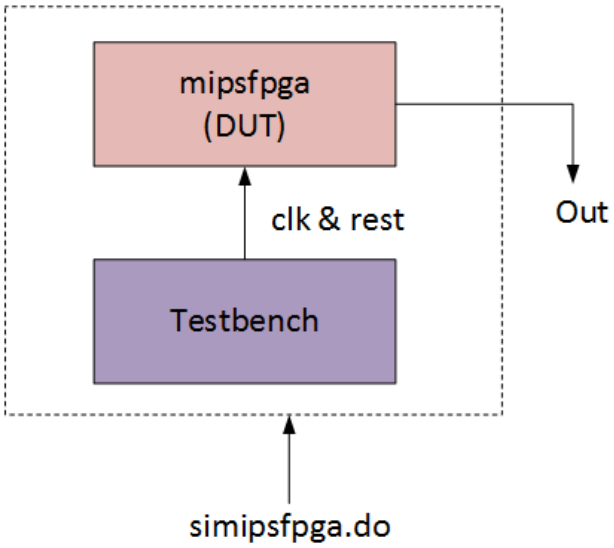

Figure 3.1: Multiplication Simulation

3.2 Synthesis

We use Altra Quartus 14.1 to target the Cyclone IV E device. The tool provides us information regarding the resource utilization of the design as well as the minimum clock period required for operation/ the maximum operating frequency.

Chapter 4

Results

4.1 Simulation

Figure 4.1 depicts the simulation results of multiply instruction. The opcode used for the generation of the same is:
2408000a - Load 0XA to register 8
24090014 - Load 0X14 to register 9
01283818 - Multiply registers 8 and 9
00003812 - Move result to register 7

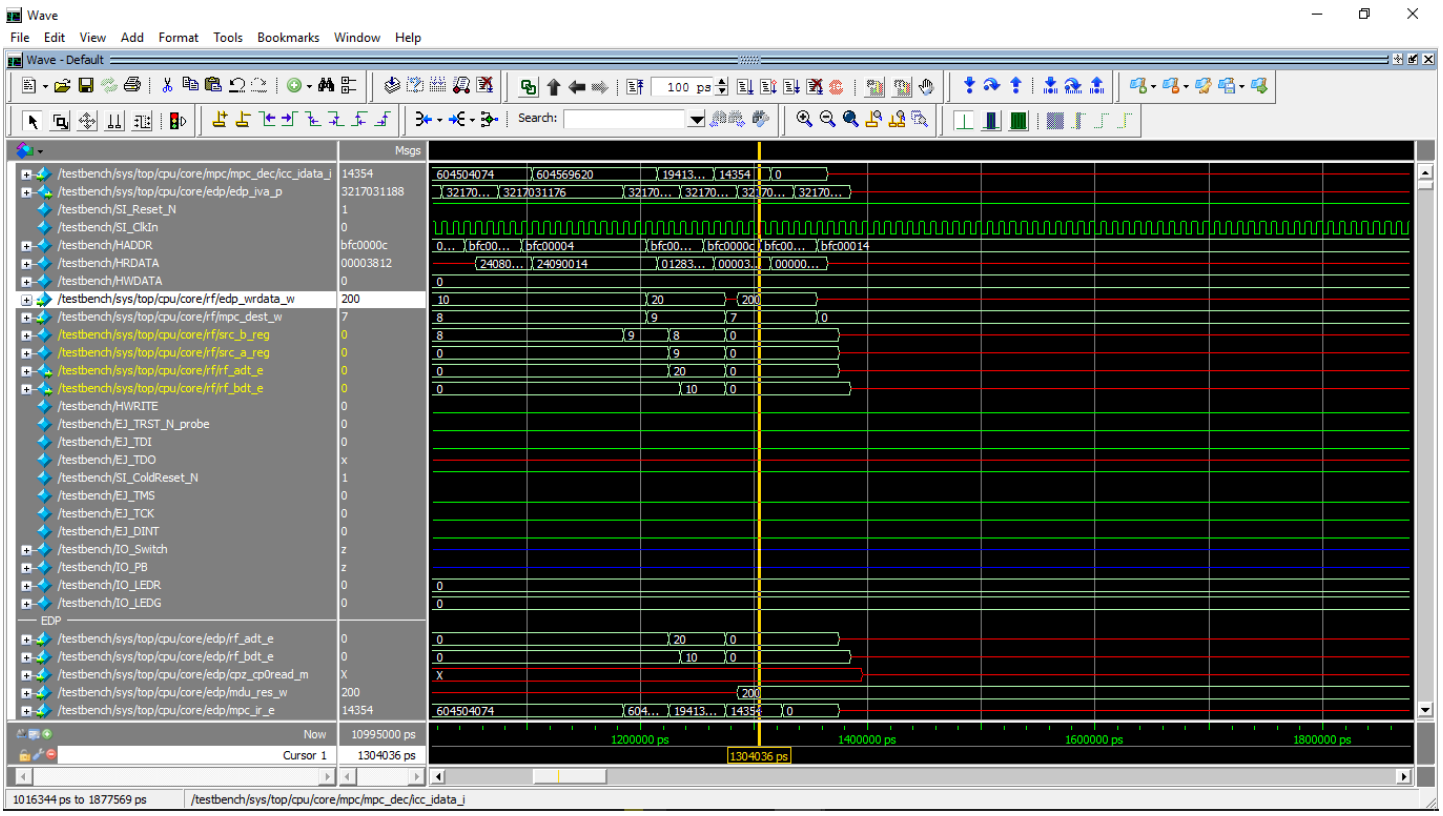

Figure 4.1: Multiplication Simulation

The signals *rf_adt_e* and *rf_bdt_e* contain the operands for multiplication. The result is obtained in *edp_wr_w* which also contains the operands on which multiplication has to be performed. The signal *mpc_dest_w* indicates the general purpose registers which hold the operands.

4.2 Synthesis

Table 4.1 shows the resource utilization of original and optimized design. From the table we can observe that, there is $\approx 70\%$ reduction in number of logic elements as compared to the original.

Table 4.1: Resource Utilization

Component	Original (No. of Logic Elements)	Optimized (No. of Logic elements)	Percentage Reduction
MMU	3692	0	100%
CPZ	1699	1406	17.25%
RF	1043	104	90%
MDU	762	251	67.1%
EDP	3239	1837	43.3%
MPC	993	862	13.2%
BIU	919	20	97.8%
DCC	1707	57	96.7%
ICC	910	55	94%
Total	14884	4592	69%

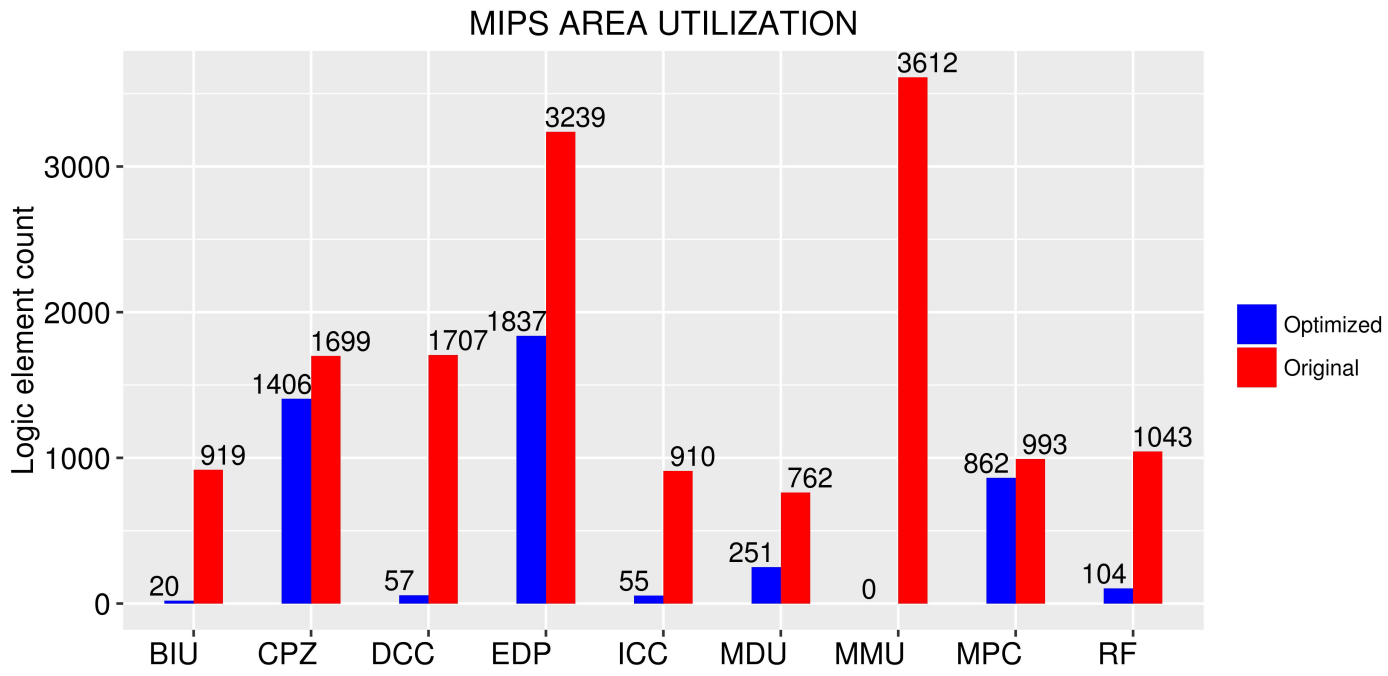

Figure 4.2: Resource Utilization of various Components

Chapter 5

Future scope & Applications

5.1 Applications

A massively parallel processor like the 256-core mapped MIPS can be used for real time signal processing applications like image processing or speech processing. The availability of special operations like multiplication and addition makes it suitable for such applications.

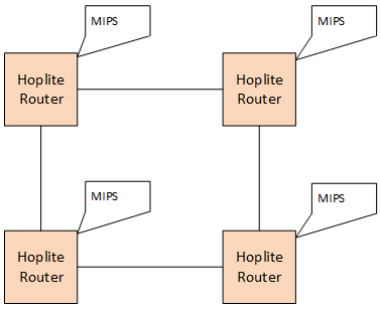

Figure 5.1: Multi core MIPS array interconnected with NOC

Chapter 6

Bibliography

References

Chih-Yuan Lien. An efficient denoising architecture for removal of impulse noise in images. URL ieeexplore.ieee.org/iel5/12/6471716/06122015.pdf.