Report

1. Introduction

In the midterm we were given the task to elaborate the specifications of the MSDAP using the behavioral model in Verilog. In homework 8, we move further defining our RTL level design and architecture and matching our specifications accordingly. In this report we describe our design of the MSDAP through the following diagram.

The various blocks are explained and elaborated accordingly.

2. MSDAP Architecture

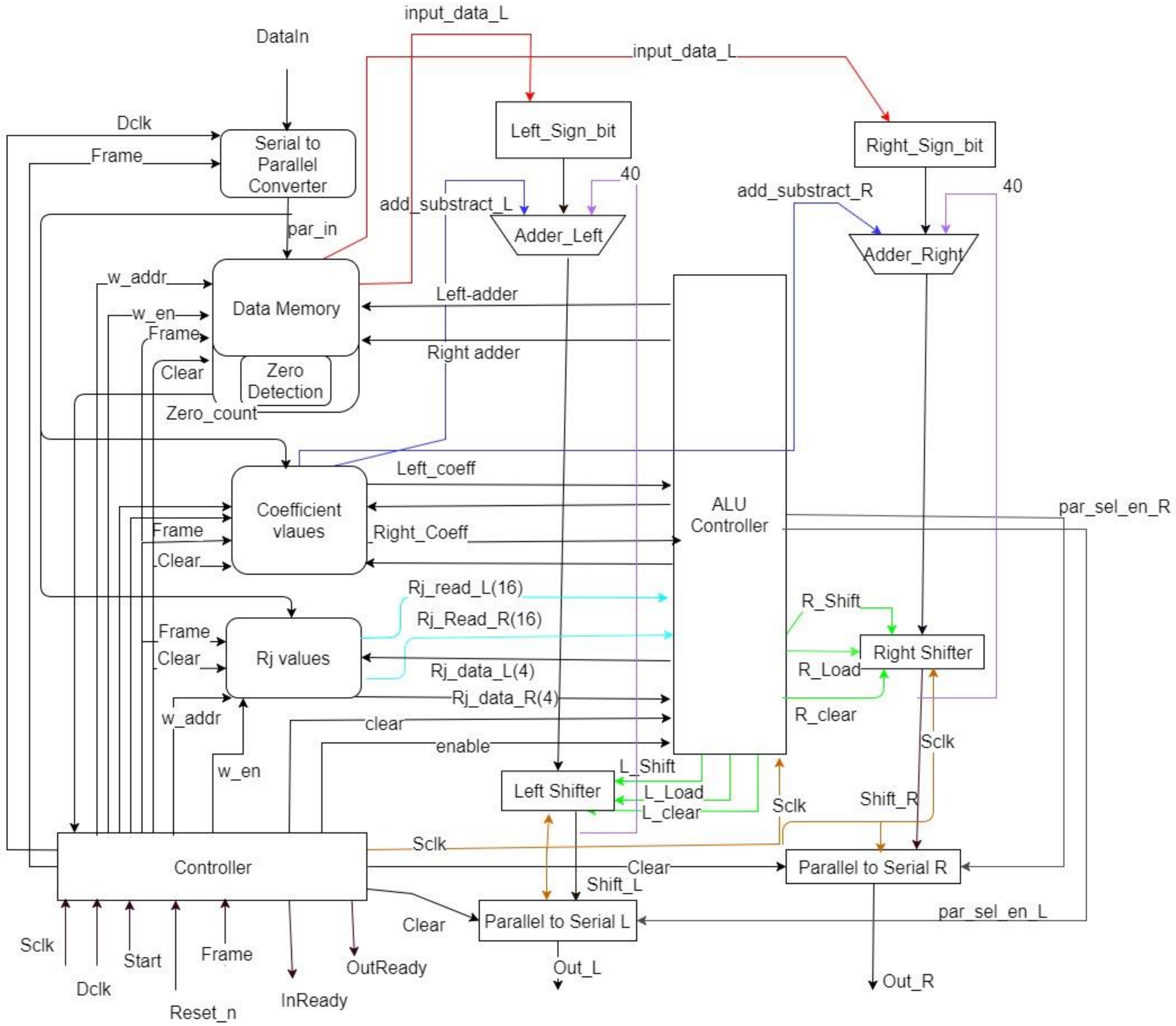


Fig -1: Input/Output Specifications with the MSDAP block diagram

3. Block Function Description

The functional blocks in our architecture are-

a. Main controller
b. Data memory
c. Mem coeff
d. Mem Rj
e. ALU Controller
f. Serial to Parallel Converter
g. Parallel to Serial Block
h. Adder/Subtract unit

We move on by descriging each functional block from the lower levels to the highest level of the main controller.

Memory Block

As shown above the Memory block of our MSDAP is divided I three important input memories for computation namely- Rj Memory, Co-efficient Memory and the Data Memory.

The distinguished functionalities of each block can be defined a follows-

Functionalities

Rj Memory- It is 16 bits wide and use to store 16 locations. Rj block is used to save Rj values. This block is defined for the two channels – Left and Right channel Rj samples. Data is written into the memory at the positive edge of the Sclk based on the address and write enable. The read address signal enables for the data to be read out. The clear signal from the FSM controller is not sensitive to this block.

Input Signals related to Rj Memory-

- write_enable - used by main controller to enable writing to memory.
- read_enable – used by ALU to enable reading from memory.
- Sclk – This is the timing reference for rj memory
- in_data – 16 bit input word from S2P. Based on write signals , at positive edge of Sclk it is written into rj memory.

- W_addr - Address is 4 bits wide coming from the MSDAP controller. It is used to write the data into the block for both right and left channel.
- Rjread - It is 4 bit address from the ALU controller used to fetch the corresponding data from the left and right rj memory.

Output Signals related to Rj Memory-

- data_rj - data_rjL, data_rjR is the 16 bit data from the left and right Rj memory respectively and is fed to the left and right ALU Control Block. The data is read when the Frame goes high.
- in_flag - Signals main controller that input is ready for processing.

Coefficient Memory- The data is being written based on the address and write enable, into the memory at the positive edge of the Sclk. Data is read out for the computation by using read address signal. It is a 16 bit address wide and has a depth of 512 locations. Both left and right channel values have one block each. 9 bit address is required.

Input Signals related to Co-efficent Memory-

- write_enable - used by the main controller to enable writing to the memory
- read_enable - used by the ALU to enable reading from the memory
- Sclk – Timing reference for coeff memory.
- in_data - 16 bit input word from the S2P. Based on the address and the write enable signals, the in_data is written to either coeff memory at the positive edge of the Sclk.
- coeffwrite - Address is 9 bits wide coming from the MSDAP controller used to write the data into the block. There are two signals for both right and left channel.
- Coeffread –9 bit address from the ALU controller used to fetch the corresponding data from the left and right coeff memory.

Output Signals related to Co-efficent Memory-

Data_coeff - data_coeffL, data_coeffR is the 16 bit data from the left and right coeff memory respectively and is fed to the left and right ALU Control Block. The data is read when the Frame goes high.

Data Memory- Upon activation of write enable signal and address, each time a Frame is received by the SIPO from the FSM controller the data is being written to the Data block at the positive edge of the Sclk. Data block is 16 bits wide and a depth of 256 locations.

It acts like a circular / cyclic buffer since once the Data block is filled, it is re-used for the new incoming data from the 0th location. 2 such data blocks for left and right channel input data samples. All the data in the Data Block memory is cleared when a clear signal is sent by the FSM controller. 8 bits address is required.

Input Signals related to Data Memory-

- write_enable - Write enable is used by the main controller to enable writing to the memory
- read_enable - used by the ALU to enable reading from the memory
- Sclk - timing reference for the data memory.
- In_data - 16 bit input word from the S2P. Based on the address and the write enable signals, the data_in is written to either data memory at the positive edge of the Sclk.
- Datawrite- Address is 8 bits wide coming from the MSDAP controller. It is used to write the data into the block. There are two signals for both right and left channel.
- Dataread - 8 bit address from the ALU controller. This is used to fetch the corresponding data from the left and right data memory.

Output signals related to Data Memory

- input_data - input_dataL, input_dataR is the 16 bit data from the left and right data memory respectively and is fed to the left and right ALU Control Block. The data is read when the Frame goes high.
- in_flag - signals the main controller that the input is ready for processing
- flag_zero - used to enforce the 800 zeroes condition for sleep mode

ALU Controller:

Functionality:

This block receives the required signals from control block and the data from the mem_coeff and mem_rj. When the need for computation occurs, the ALU controller gets activated.

Input Signals relate to the ALU

The inputs to this block are – enable, clear,

- Enable – This signal from the Main controller activates the ALU controller. When this 1bit signal is high, the computation begins on the input sample. Clear – This is required to reset the data and clear the data of the accumulator. Rj_read_l, rj_read_r – This is the 16 bit data from the mem_rj to be fed to the alu_controller. When enable is on, the data is fed. Coeff_L, coeff_R – This 16 bit coefficient value is fed to the ALU controller when the enable is high Sclk – This is the timing reference signal to the ALU controller to perform operations like load and shift.
- Sleep – This flag tells the ALU controller that the main controller is in sleep mode. This will make the ALU controller operations to come to a halt.

Output Signals relate to the ALU

The outputs of the ALU Controller are-

- Rj_data_L, Rj_data_R – This is the 4 bit address of the blocks.
- Coeff_data_L, coeff_data_R – This is the 9 bit address of the coefficient blocks according to the rj value.
- L_addr, R_addr – This is an 8 bit read address to get the input data from the channels. When this address is given to the mem block, data is accessed.
- L_load, R_load – This bit shows that the accumulator has the value 0 loaded to it. This load signal is aligned to Frame.
- L_Clear ,R_Clear – The 40 bit wide register is to be made zero.
- L_shift, R_shift – This indicates that the shift operation has to be done on the left and right adders.
- Par_sel_en_l, par_sel_en_R – This is the signal given to the PISO so that the block converts the output.

Parallel to Serial L/R Block:

Functionality: When Sclk is on its rising edge, the Parallel to Serial Converter functions. The parallel data input to this block is the data input beginning from the MSB (16 bits) at the beginning of the frame and rising edge of Sclk. The MSB is received when the frame signal changes to high and the remaining 15 bits comes at every rising edge of the Sclk.

Input signals relate to the Parallel to Serial L/R Block

The inputs to this block are Clear, Sclk, Shift_L, Shift_R, par_ser_en_L, par_ser_en_R.

- Clear is synchronous with Sclk. When the clear signal is high,data in this block gets cleared.
- Sclk – the data is received when the Sclk signal is high. This is the timing reference for the incoming parallel data input.
- Shift_L, Shift_R – This is the parallel inputs given to the block. This input is received at the positive edge of the Sclk. The input comes from the output of Shifter_1_L/Shifter_1_R
- par_ser_en_L, par_ser_en_R – When the output is ready and has to be given out in a serial manner, the ALU sets this bit as high. It is a 1 bit input given to the block from the ALU.

Output Signals related to the Parallel to Serial L/R Block

The output of this block is Out_L, Out_R.

- Out_L, Out_R – This is the serial output from the Parallel to Serial converter block.

Serial to Parallel Converter:

Functionality:

When Dclk is on its rising edge, the Serial to Parallel Converter functions. The input to this block is the Serial data input beginning from the MSB (16 bits) at the beginning of the frame. The MSB is received when the frame signal changes to high and the remaining 15 bits comes at every rising edge of the Dclk. When this serial data becomes combined into a 16 bit word, the Serial to Parallel Converter indicates to the Main Controller to enable data memory to transfer the address.

Input Signals relate to the Serial to Parallel converter

The inputs of this block are Frame, Clear, Indata.

- Frame- This works with Dclk and when the first bit of the word is read, this is high. After the first Dclk cycle, the frame goes low.

- Clear- This input is used to remove all the data present in the block whenever the main controller issues a reset.
- Dclk – This is the timing reference for receiving the input data in a serial manner. The data is received at the positive edge of the Dclk
- Indata – This is the serial input to the block. The data is received at the positive edge of the Dclk. The MSB is transmitted first and LSB last.

Output Signals related to Serial to Parallel Converter

The output is the par_in

- Par_in- This is the data (16 bits) which is received from the block as output and is fed to the memory.

Main Controller:

Functionality

Main controller manipulates the control signals for both channels. At a lower layer, the ALU block is controlled by the main controller.

Input Signals related to the Main Controller

- Sclk - system clock that is used to synchronize all the control signals and IO signals. With the negative or falling edge of the System clock, all the state changes and the computation happen. The frequency of the Sclk determines the frequency of operation of the entire system/ chip.
- Dclk - Dclk is the Data clock that is used as the timing reference to receive and send the input samples. At the falling edge of the Dclk, a bit is received or sent. Hence to receive a 16 bit data, 16 Dclk cycles are required. One Dclk is 35 Sclk cycles.
- Frame – While sending or receiving the input or output to or from the MSDAP, the Frame is set high. Frame signifies the start of the first bit of data-input, output, Rj or co-efficient. Frame is used as active high signal and is made high for one Dclk cycle. After receiving the first bit of the data, the Frame goes inactive. It is active after every 16 Dclk cycles and hence a 16 bit data is received between two successive Frames.

- Start - An asynchronous input signal from the Controller to MSDAP used to indicate the start of the FSM and initialization of chip begins. Until then the chip is said to be idle. Start is an active high input signal.
- Reset_n - Reset is an active low asynchronous input signal to the MSDAP. The chip goes to the reset mode when this signal is applied.
- In_flag - Signals to the controller that the input data is ready for processing.
- Flag_zeroL and flaf_zeroR - Signals from the data memory block that indicate that there have been 800 zeroes detected and both flags are ANDed to generate sleep_flag.

Output signals related to the Main Controller

- Rjwrite, coeffwrite, datawrite - These signals are used for signaling the memories of the addresses where the incoming data must be written.
- Rj_enable, data_enable, coeff_enable - Enables for routing the data to the correct memories as it comes in
- Frame_out, Dclk_out, Sclk_out - Internal signals for Frame, Dclk and Sclk, distributed from the main controller
- Work_enable - Indicates to the ALU that a computation iteration can begin
- Sleep_flag - Signals to the ALU that the main controller will be in sleep mode after 800 continuous zeroes and that computations must cease.
- InReady - InReady is an output signal to the Controller from the MSDAP which is set high when the chip is ready to receive the data from both the channels. InReady is aligned with the Frame.

Adder Block:

Functionality:

Addition/Subtraction is performed in this block depending on the add_sub_L or add_sub_R signal from the coefficient memory block. The main aim for this is to reduce the huge architecture issues which would occur if multiplications where used instead of addition or shifting. The total number of additions or subtractions in this unit is 512 for one computation of the input data sample. Addition/subtraction takes place at the positive edge of the Sclk. One of the operands come from the sign extension block and the other operand comes from the Shifter block. Both operands are 40 bits wide. The result which is 40 bits is fed back to the Shifter block.

Input Signals related to the Adder Block

- Addsub - addsubL is 1 bit wide coming from the ALU controller based on the MSB of the coefficient value read from memory. This bit decides whether addition or subtraction that has to be performed between the operands of the adder_L.
- A - a is 40 bits wide which is sign extended and padded version of xin_data_L coming from the sign_pad_L block. This is one of the operands for adder_L
- B - b is 40 bits wide which is fed as another operand to the adder_L block.
- Adder_en - adder_en is used to enable the adder to perform addition/subtraction at required times

Output Signals related to the Adder Block

- Sum- This is the result of a and b which is 40 bits wide. This data is fed to the shift_acc_L block.

Shift and Accumulate block:

Functionality:

The content in this block is initially set to 40 bit zero. It is mainly used to follow the efficient algorithm that was planned for the MSDAP. It holds the result of the adder block when a load signal from the ALU control block is applied to it. One bit is right shifted whenever shift_enable signal from the ALU controller is applied. Shifting is done in the positive edge of the Sclk. Whenever there is a Reset signal from the ALU control block, the contents in the accumulator is cleared.

Input Signals related to the Shifter Block

- Adder_out- To ensure the correct logic of the algorithm of the MSDAP this signal is need for the 1 bit linear shifting.

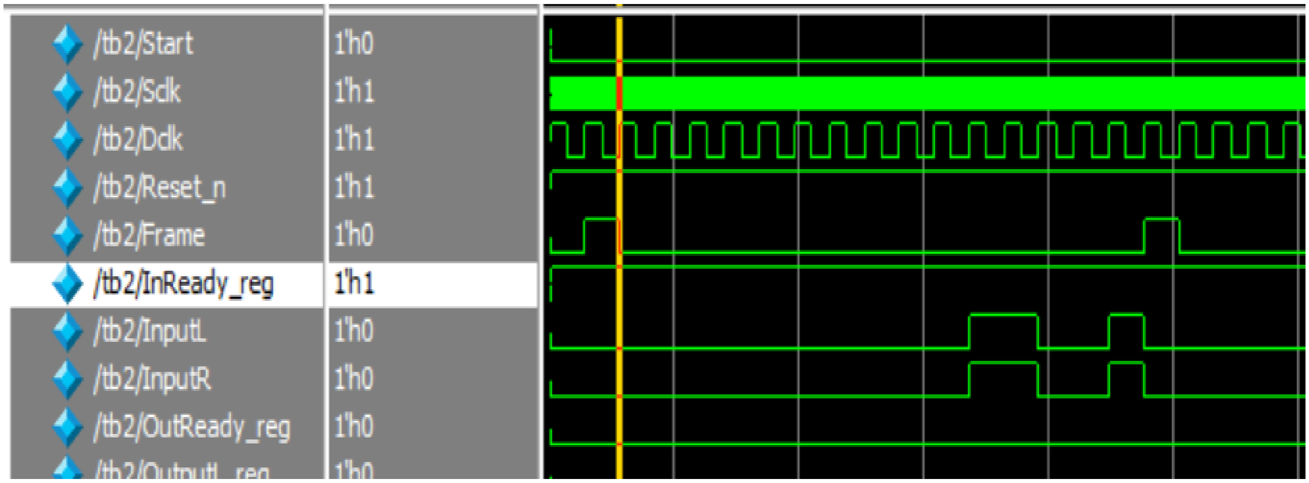
Output Signals relate to the Shift Block

- The shiftout_L or shiftout_R is 40 bits which is fed back as one of the operand to the respective adder block.

4. Testbench and Simulation Results

To ensure that our architecture matches with the verified behavior of the MDAP we present below the Modelsim simulation of our structural model Verilog code.

Matching the Input Signal Specifications

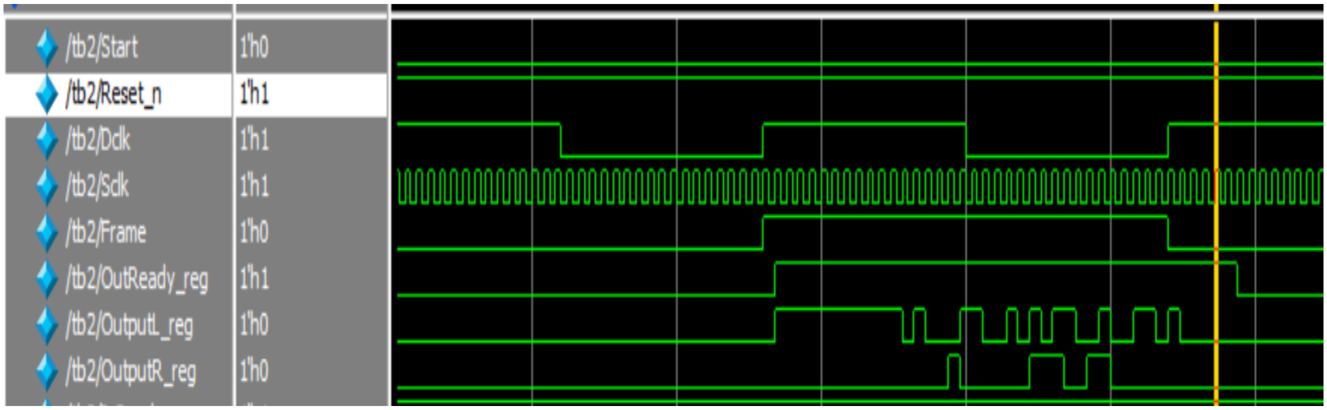


/tb2/Start	1'h0
/tb2/Sclk	1'h1
/tb2/Dclk	1'h1
/tb2/Reset_n	1'h1
/tb2/Frame	1'h0
/tb2/InReady_reg	1'h1
/tb2/InputL	1'h0
/tb2/InputR	1'h0
/tb2/OutReady_reg	1'h0
/tb2/OutputL_reg	1'h0

Fig2: Input signals specified in previous project for Behavioral Model matching with the testbench of the Structural model

Showing that each input is being read in 1 Frame cycle.

Matching the Output Signal Specifications

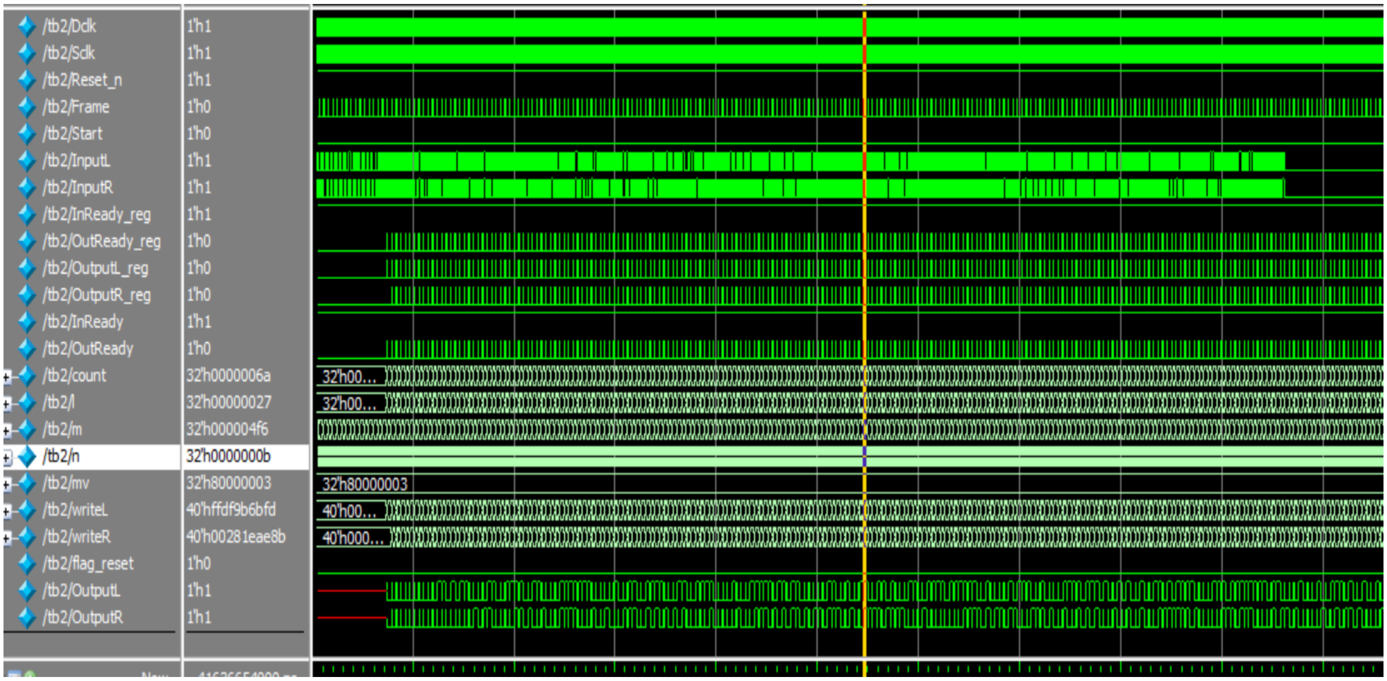


/tb2/Start	1'h0
/tb2/Reset_n	1'h1
/tb2/Dclk	1'h1
/tb2/Sclk	1'h1
/tb2/Frame	1'h0
/tb2/OutReady_reg	1'h1
/tb2/OutputL_reg	1'h0
/tb2/OutputR_reg	1'h0

Fig3 : Output of the Structural Verilog code

Verification and matching of the Output waveforms from Midterm Project can be done seeing that on the rising edge of the Sclk, with the Frame and OutReady going high we start getting outputs.

Overall Simulation Results



/tb2/Dclk	1'h1
/tb2/Sclk	1'h1
/tb2/Reset_n	1'h1
/tb2/Frame	1'h0
/tb2/Start	1'h0
/tb2/InputL	1'h1
/tb2/InputR	1'h1
/tb2/InReady_reg	1'h1
/tb2/OutReady_reg	1'h0
/tb2/OutputL_reg	1'h0
/tb2/OutputR_reg	1'h0
/tb2/InReady	1'h1
/tb2/OutReady	1'h0
/tb2/count	32'h0000006a
/tb2/l	32'h00000027
/tb2/m	32'h000004f6
/tb2/n	32'h0000000b
/tb2/mv	32'h80000003
/tb2/writeL	40'hffdf9b6bfd
/tb2/writeR	40'h00281eae8b
/tb2/flag_reset	1'h0
/tb2/OutputL	1'h1
/tb2/OutputR	1'h1

Fig4: Elaborate Signal Description via testbench

Although the diagram is not so clear, we present here almost all the important signals that made our architecture function properly and generate correct outputs.

5. Conclusion

We are confident that the architecture design that we presented above is accurate for the MSDAP's behavior and can be placed and routed accordingly.