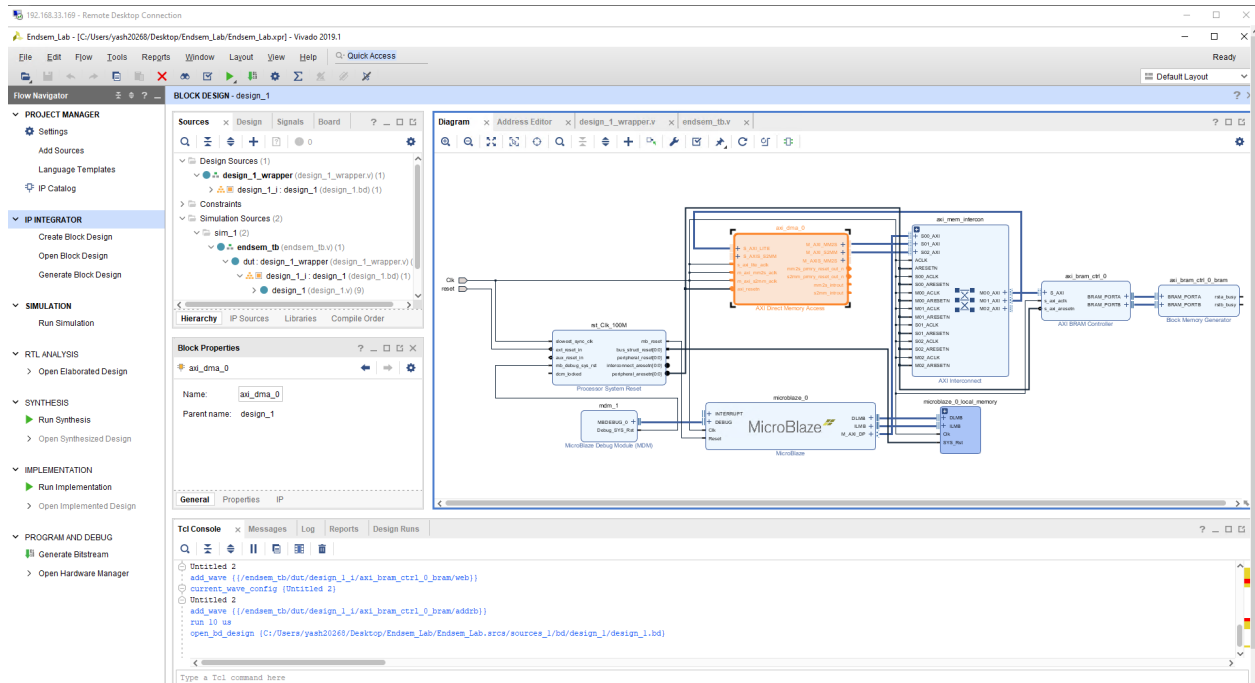
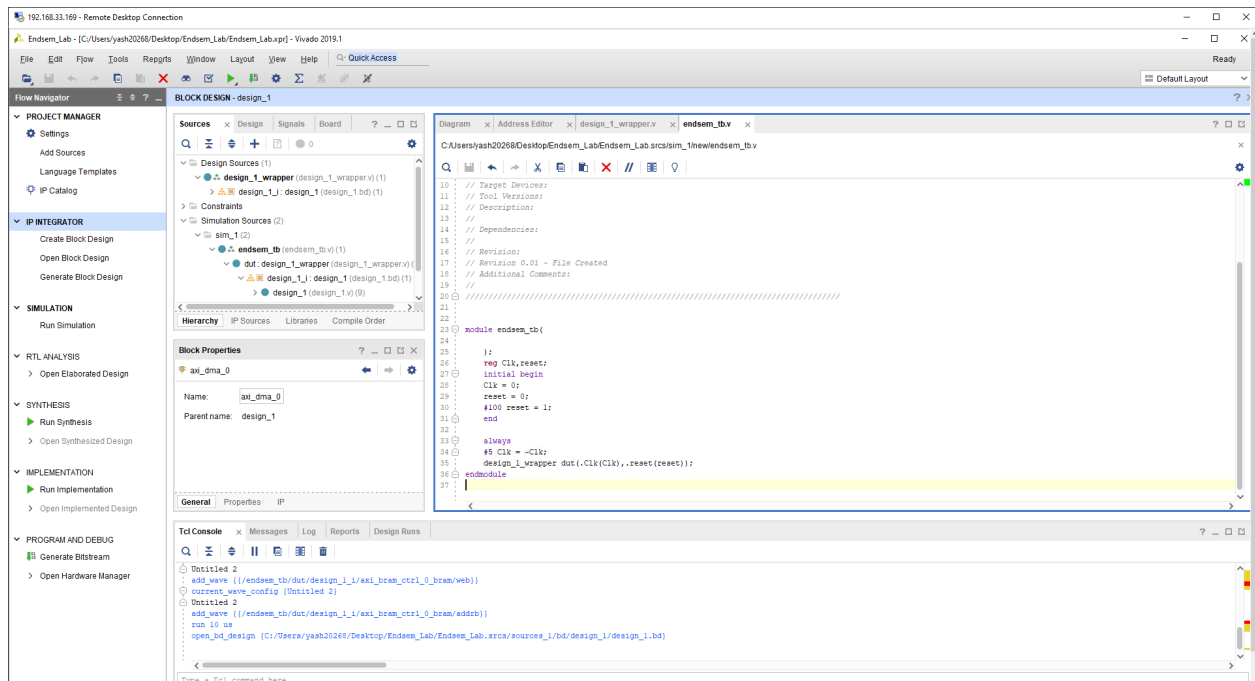


# ELD ENDSEM LAB

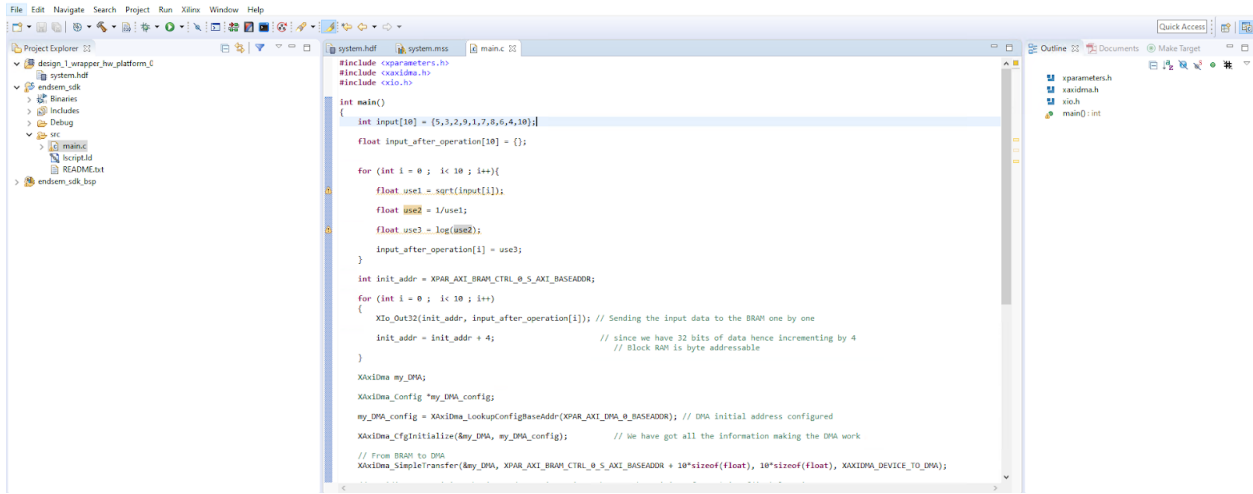
## BLOCK DIAGRAM



## TEST BENCH



# SDK CODE



```
File Edit Navigate Search Project Run Xilinx Window Help
Project Explorer
design_1_wrapper_hw_platform_0
system.mss
endem_sdk
  Binaries
  Includes
  Debug
  src
    main.c
    hcrptId
    README.txt
endem_sdk_top

system.mss
main.c
#include <parameters.h>
#include <xaxidma.h>
#include <xio.h>

int main()
{
    int input[10] = {5,3,2,9,1,7,8,6,4,10};
    float input_after_operation[10] = {};

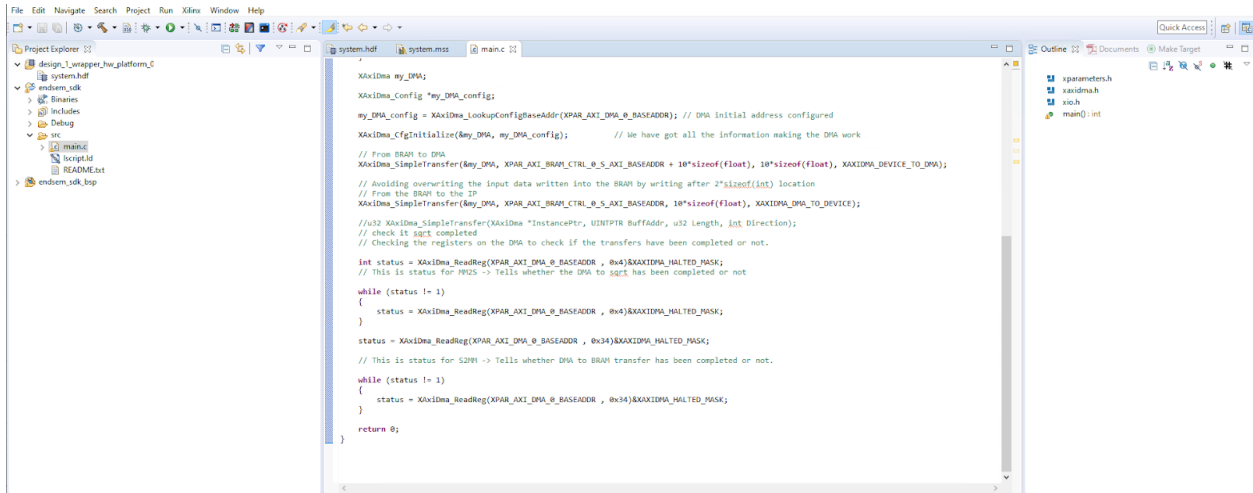
    for (int i = 0 ; i< 10 ; i++){
        float use1 = sqrt(input[i]);
        float use2 = 1/use1;
        float use3 = log(use2);

        input_after_operation[i] = use3;
    }

    int init_addr = XPAR_AXI_BRAM_CTRL_0_S_AXI_BASEADDR;
    for (int i = 0 ; i< 10 ; i++)
    {
        Xio_Out32(init_addr, input_after_operation[i]); // Sending the input data to the BRAM one by one
        init_addr = init_addr + 4; // since we have 32 bits of data hence incrementing by 4
        // Block RAM is byte addressable
    }

    XaxiDma my_DMA;
    XaxiDma_Config *my_DMA_config;
    my_DMA_config = XaxiDma_LookupConfigBaseAddr(XPAR_AXI_DPA_0_BASEADDR); // DPA Initial address configured
    XaxiDma_CfgInitialize(&my_DMA, my_DMA_config); // We have got all the information making the DPA work

    // From BRAM to DPA
    XaxiDma_SimpleTransfer(&my_DMA, XPAR_AXI_BRAM_CTRL_0_S_AXI_BASEADDR + 10*sizeof(float), 10*sizeof(float), XAXIDMA_DEVICE_TO_DPA);
}
```



```
File Edit Navigate Search Project Run Xilinx Window Help
Project Explorer
design_1_wrapper_hw_platform_0
system.mss
endem_sdk
  Binaries
  Includes
  Debug
  src
    main.c
    hcrptId
    README.txt
endem_sdk_top

system.mss
main.c
XaxiDma my_DMA;
XaxiDma_Config *my_DMA_config;
my_DMA_config = XaxiDma_LookupConfigBaseAddr(XPAR_AXI_DPA_0_BASEADDR); // DPA Initial address configured
XaxiDma_CfgInitialize(&my_DMA, my_DMA_config); // We have got all the information making the DPA work

// From BRAM to DPA
XaxiDma_SimpleTransfer(&my_DMA, XPAR_AXI_BRAM_CTRL_0_S_AXI_BASEADDR + 10*sizeof(float), 10*sizeof(float), XAXIDMA_DEVICE_TO_DPA);
// Avoiding overwriting the input data written into the BRAM by writing after 2*sizeof(int) location
// From the BRAM to the IP
XaxiDma_SimpleTransfer(&my_DMA, XPAR_AXI_BRAM_CTRL_0_S_AXI_BASEADDR, 10*sizeof(float), XAXIDMA_DPA_TO_DEVICE);
//u32 XaxiDma_SimpleTransfer(XaxiDma *InstancePtr, UINTPTR BuffAddr, u32 Length, int Direction);
// check if sqrt completed
// Checking the registers on the DPA to check if the transfers have been completed or not.

int status = XaxiDma_ReadReg(XPAR_AXI_DPA_0_BASEADDR, 0x4)&XAXIDMA_HALTED_MASK;
// This is status for FPD5 -> Tells whether the DPA to sqrt has been completed or not

while (status != 1)
{
    status = XaxiDma_ReadReg(XPAR_AXI_DPA_0_BASEADDR, 0x4)&XAXIDMA_HALTED_MASK;
}

status = XaxiDma_ReadReg(XPAR_AXI_DPA_0_BASEADDR, 0x34)&XAXIDMA_HALTED_MASK;
// This is status for S2D4H -> Tells whether DPA to BRAM transfer has been completed or not.

while (status != 1)
{
    status = XaxiDma_ReadReg(XPAR_AXI_DPA_0_BASEADDR, 0x34)&XAXIDMA_HALTED_MASK;
}

return 0;
}
```

# WAVE FORM

