Lab6

M23EEV019

Suraj Prajapati

Block Design on PynqZ2

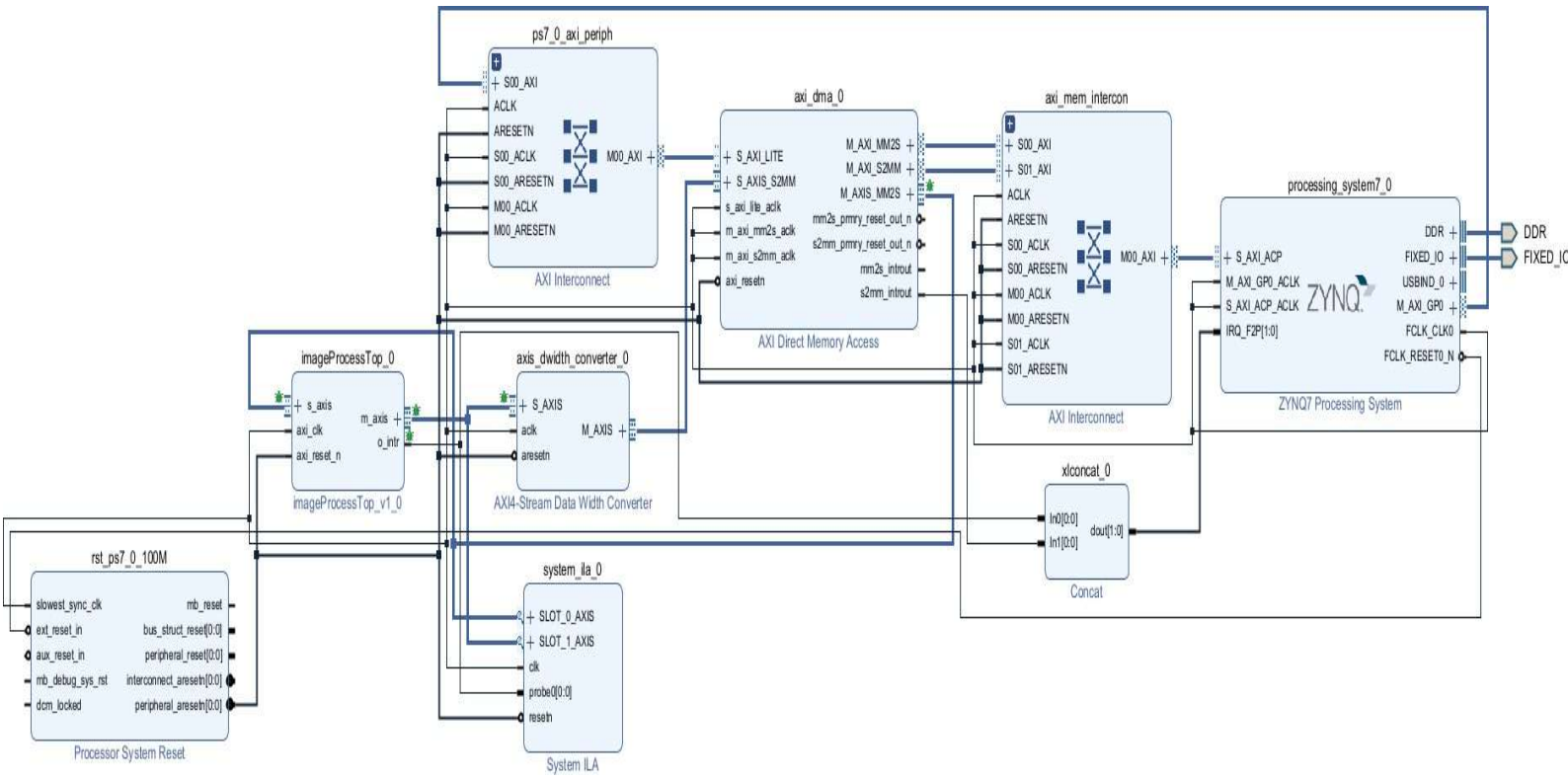


Image Preprocessing

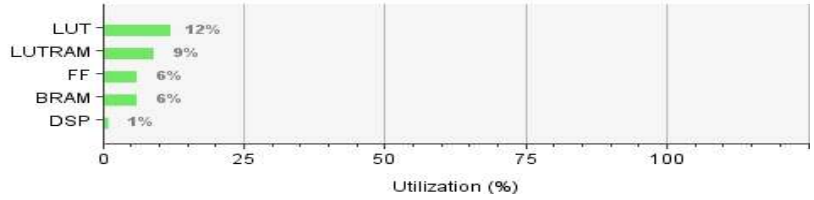
Converting any dimension image to 512*512 bmp(bitmap) image which then goes into simulation in vivado .

```python
##################### Preprocessing
from PIL import Image

def convert_to_grayscale(image_path, output_path):

    image = Image.open(image_path)


    grayscale_image = image.convert("L")


    resized_image = grayscale_image.resize((512, 512))
```

```
    resized_image.save(output_path)


input_image_path = "n1.jpg"
output_image_path = "output_image.bmp"
convert_to_grayscale(input_image_path, output_image_path)
```

Utilization

Resource	Utilization	Available	Utilization %
LUT	6271	53200	11.79
LUTRAM	1587	17400	9.12
FF	6543	106400	6.15
BRAM	8	140	5.71
DSP	2	220	0.91

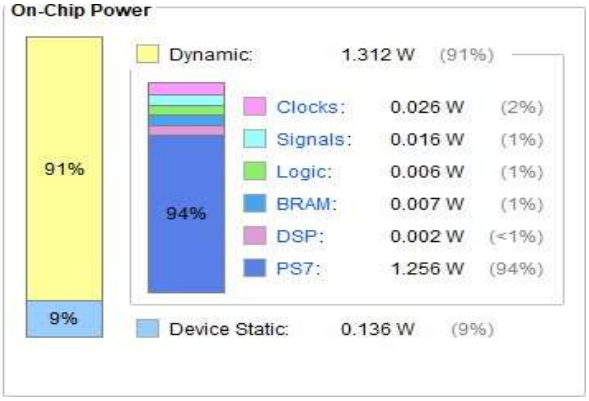


Power

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**1.449 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**41.7°C**
Thermal Margin:	43.3°C (3.6 W)
Effective ϑJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Medium

Launch Power Constraint Advisor to find and fix invalid switching activity



Vitis Code

```
#include "xaxidma.h"
#include "xparameters.h"
#include "sleep.h"
#include "xil_cache.h"
#include "xil_io.h"
#include "xscugic.h"
#include "imageData.h"
```

```
#include "xuartps.h"

#define imageSize 512*512

u32 checkHalted(u32 baseAddress,u32 offset);

XScuGic IntcInstance;
static void imageProcISR(void *CallBackRef);
static void dmaReceiveISR(void *CallBackRef);
int done;

int main(){
    u32 status;
      u32 totalTransmittedBytes=0;
      u32 transmittedBytes = 0;
      XUartPs_Config *myUartConfig;
      XUartPs myUart;

      //Initialize uart
      myUartConfig = XUartPs_LookupConfig(XPAR_PS7_UART_0_DEVICE_ID);
      status = XUartPs_CfgInitialize(&myUart, myUartConfig,
myUartConfig->BaseAddress);
      if(status != XST_SUCCESS)
            print("Uart initialization failed...\n\r");
      status = XUartPs_SetBaudRate(&myUart, 115200);
      if(status != XST_SUCCESS)
            print("Baudrate init failed..... \n\r");

      XAxiDma_Config *myDmaConfig;
      XAxiDma myDma;
    //DMA Controller Configuration
      myDmaConfig = XAxiDma_LookupConfigBaseAddr(XPAR_AXI_DMA_0_BASEADDR);
      status = XAxiDma_CfgInitialize(&myDma, myDmaConfig);
      if(status != XST_SUCCESS){
            print("DMA initialization failed\n");
            return -1;
      }

      XAxiDma_IntrEnable(&myDma, XAXIDMA_IRQ_IOC_MASK, XAXIDMA_DEVICE_TO_DMA);

      //Interrupt Controller Configuration
      XScuGic_Config *IntcConfig;
      IntcConfig = XScuGic_LookupConfig(XPAR_PS7_SCUGIC_0_DEVICE_ID);
      status =  XScuGic_CfgInitialize(&IntcInstance, IntcConfig,
IntcConfig->CpuBaseAddress);

      if(status != XST_SUCCESS){
```

```
		xil_printf("Interrupt controller initialization failed..");
		return -1;
	}


XScuGic_SetPriorityTriggerType(&IntcInstance,XPAR_FABRIC_IMAGEPROCESSTOP_0_O_INTR_I
NTR,0xA0,3);
	status =
XScuGic_Connect(&IntcInstance,XPAR_FABRIC_IMAGEPROCESSTOP_0_O_INTR_INTR,(Xil_Interr
uptHandler)imageProcISR,(void *)&myDma);
	if(status != XST_SUCCESS){
		xil_printf("Interrupt connection failed");
		return -1;
	}
	XScuGic_Enable(&IntcInstance,XPAR_FABRIC_IMAGEPROCESSTOP_0_O_INTR_INTR);


XScuGic_SetPriorityTriggerType(&IntcInstance,XPAR_FABRIC_AXI_DMA_0_S2MM_INTROUT_INT
R,0xA1,3);
	status =
XScuGic_Connect(&IntcInstance,XPAR_FABRIC_AXI_DMA_0_S2MM_INTROUT_INTR,(Xil_Interrup
tHandler)dmaReceiveISR,(void *)&myDma);
	if(status != XST_SUCCESS){
		xil_printf("Interrupt connection failed");
		return -1;
	}
	XScuGic_Enable(&IntcInstance,XPAR_FABRIC_AXI_DMA_0_S2MM_INTROUT_INTR);

	Xil_ExceptionInit();

Xil_ExceptionRegisterHandler(XIL_EXCEPTION_ID_INT,(Xil_ExceptionHandler)XScuGic_Int
erruptHandler,(void *)&IntcInstance);
	Xil_ExceptionEnable();


	status = XAxiDma_SimpleTransfer(&myDma,(u32)imageData,
512*512,XAXIDMA_DEVICE_TO_DMA);
	status = XAxiDma_SimpleTransfer(&myDma,(u32)imageData,
4*512,XAXIDMA_DMA_TO_DEVICE);//typecasting in C/C++
	if(status != XST_SUCCESS){
		print("DMA initialization failed\n");
		return -1;
	}


    while(!done){
```

```
    }


       while(totalTransmittedBytes < imageSize){
              transmittedBytes =
XUartPs_Send(&myUart,(u8*)&imageData[totalTransmittedBytes],1);
              totalTransmittedBytes += transmittedBytes;
              usleep(1000);
       }


}


u32 checkIdle(u32 baseAddress,u32 offset){
       u32 status;
       status = (XAxiDma_ReadReg(baseAddress,offset))&XAXIDMA_IDLE_MASK;
       return status;
}


static void imageProcISR(void *CallBackRef){
       static int i=4;
       int status;
       XScuGic_Disable(&IntcInstance,XPAR_FABRIC_IMAGEPROCESSTOP_0_O_INTR_INTR);
       status = checkIdle(XPAR_AXI_DMA_0_BASEADDR,0x4);
       while(status == 0)
              status = checkIdle(XPAR_AXI_DMA_0_BASEADDR,0x4);
       if(i<514){
              status = XAxiDma_SimpleTransfer((XAxiDma
*)CallBackRef,(u32)&imageData[i*512],512,XAXIDMA_DMA_TO_DEVICE);
              i++;
       }
       XScuGic_Enable(&IntcInstance,XPAR_FABRIC_IMAGEPROCESSTOP_0_O_INTR_INTR);
}


static void dmaReceiveISR(void *CallBackRef){
       XAxiDma_IntrDisable((XAxiDma *)CallBackRef, XAXIDMA_IRQ_IOC_MASK,
XAXIDMA_DEVICE_TO_DMA);
       XAxiDma_IntrAckIrq((XAxiDma *)CallBackRef, XAXIDMA_IRQ_IOC_MASK,
XAXIDMA_DEVICE_TO_DMA);
       done = 1;
       XAxiDma_IntrEnable((XAxiDma *)CallBackRef, XAXIDMA_IRQ_IOC_MASK,
XAXIDMA_DEVICE_TO_DMA);
}
```

Results

A:original 512*512 grayscale image

B:Edge detected 512*512