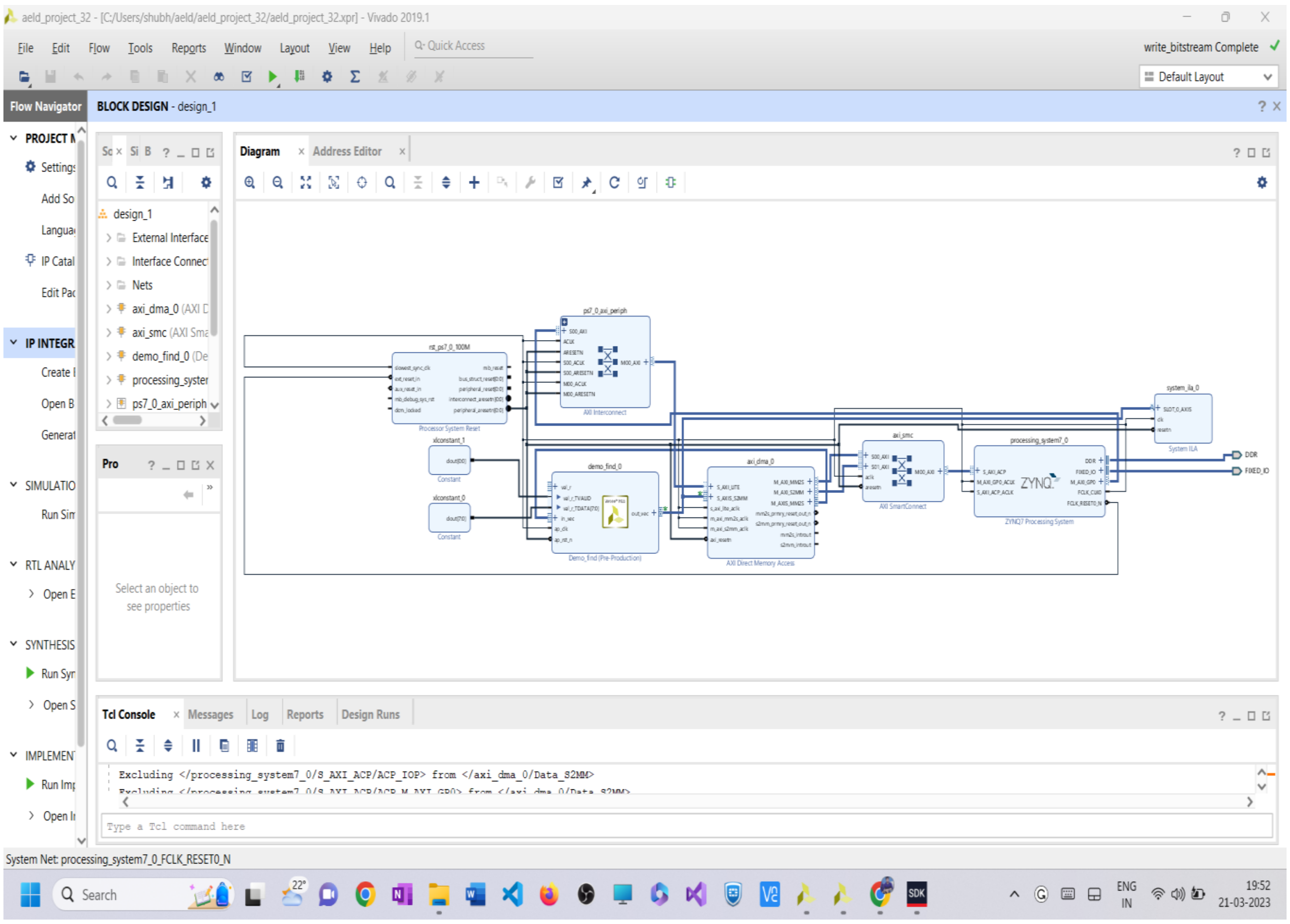
LAB ASSIGNMENT -5

Name – Aryan Gupta

Roll No – MT22154

Q1)-



```
Excluding </processing_system7_0/S_AXI_ACP/ACP_IOP> from </axi_dma_0/Data_S2MM>
```

System Net: processing_system7_0_FCLK_RESET0_N

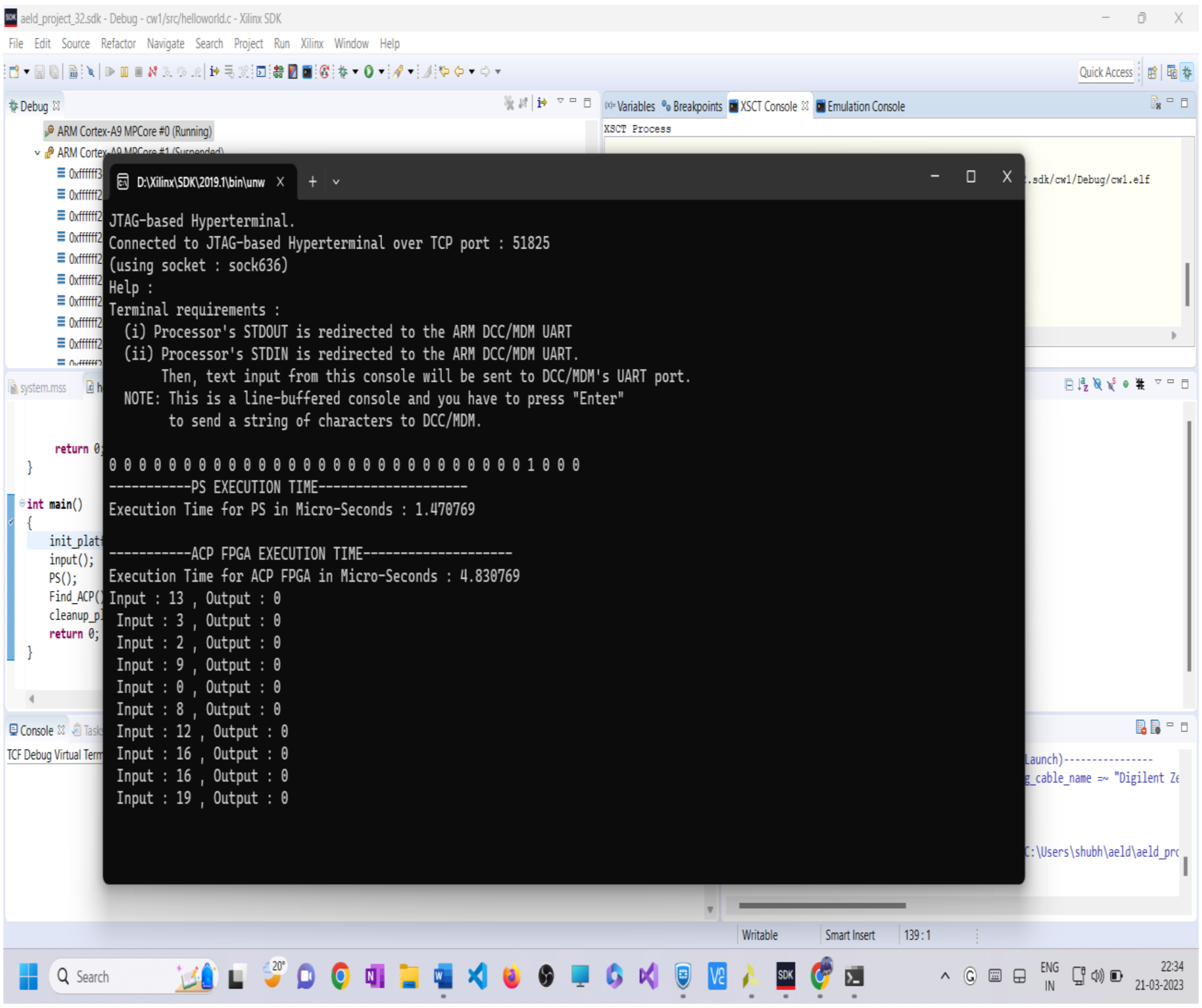
Code :-

```
/*****************************************************************************
*
*
* Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reserved.
*
* Permission is hereby granted, free of charge, to any person obtaining a copy
* of this software and associated documentation files (the "Software"), to
deal
* in the Software without restriction, including without limitation the rights
* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
* copies of the Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be included in
```

```c
* all copies or substantial portions of the Software.
*
* Use of the Software is limited solely to applications:
* (a) running on a Xilinx device, or
* (b) that interact with a Xilinx device through a bus or interconnect.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
* XILINX  BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF
* OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
* SOFTWARE.
*
* Except as contained in this notice, the name of the Xilinx shall not be used
* in advertising or otherwise to promote the sale, use or other dealings in
* this Software without prior written authorization from Xilinx.
*
******************************************************************************
/

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

#include <stdio.h>
#include <stdlib.h>
#include "xaxidma.h"
#include "xparameters.h"
#include "platform.h"
#include <xtime_l.h>
#define INP_SIZE 32
int Find_input[INP_SIZE],Find_output[INP_SIZE];
int val=4;
int find_outputps[INP_SIZE];
void input()
{
        for(int i=0;i<INP_SIZE;i++)
        {
          Find_input[i]= (rand()%20);
        }
}
void PS()
{
  XTime time_PS_start,time_PS_end;
  XTime_SetTime(0);
  XTime_GetTime(&time_PS_start);
            for(int i=0; i<INP_SIZE; i++)
          {
            if(Find_input[i]==val)
            {
              find_outputps[i]=1;
            }
            else
              find_outputps[i]=0;
          }
         XTime_GetTime(&time_PS_end);
          for(int i=0; i<INP_SIZE; i++)
          {
            printf("%d ",find_outputps[i]);
          }

          printf("\n-----------PS EXECUTION TIME--------------------
\n");
            float time_processor = 0;
            time_processor = (float)1.0 * (time_PS_end -
time_PS_start) / (COUNTS_PER_SECOND/1000000);
            printf("Execution Time for PS in Micro-Seconds : %f\n"
, time_processor);
}
int Find_ACP()
{
  int status;
   XAxiDma_Config *DMA_confptracp;
   XAxiDma AxiDMAacp;
   DMA_confptracp = XAxiDma_LookupConfig(XPAR_AXI_DMA_0_DEVICE_ID);
   status = XAxiDma_CfgInitialize(&AxiDMAacp, DMA_confptracp);
   if(status != XST_SUCCESS)
   {
      printf("ACP DMA Init Failed\t\n");
      return XST_FAILURE;
   }
       XTime time_ACP_start , time_ACP_end;
      XTime_SetTime(0);
      XTime_GetTime(&time_ACP_start);
```

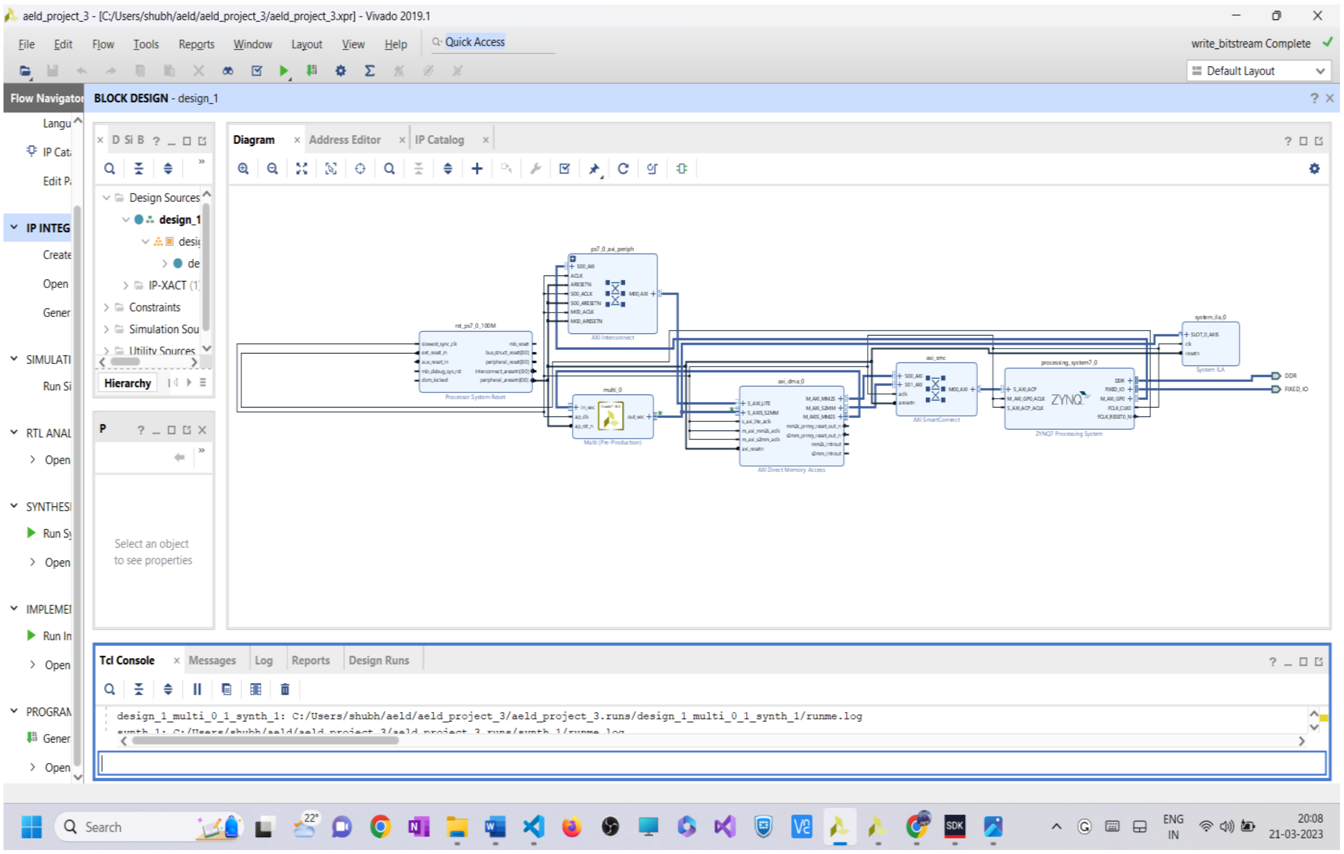
```
            // Generate random numbers

               status = XAxiDma_SimpleTransfer(&AxiDMAacp,
(UINTPTR)Find_output, (sizeof(int)*INP_SIZE),XAXIDMA_DEVICE_TO_DMA);
               status = XAxiDma_SimpleTransfer(&AxiDMAacp,
(UINTPTR)Find_input, (sizeof(int)*INP_SIZE),XAXIDMA_DMA_TO_DEVICE);
               // We have only configure the DMA to perform these two
transactions..DMA might not have started the transactions.

             //  Xil_DCacheInvalidateRange((UINTPTR)FFT_output_PLACP,
(sizeof(float complex)*FFT_Size));
               //status = checkDMAIdle(XPAR_AXI_DMA_0_BASEADDR, 0x04);
               status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x04) &
0x00000002;
               while(status!=0x00000002)
               {
                 status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x04) &
0x00000002;
               }
               status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x34) &
0x00000002;

              while(status!=0x00000002)
              {
                status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x34) &
0x00000002;
              }
              XTime_GetTime(&time_ACP_end);
                printf("\n-----------ACP FPGA EXECUTION TIME-----------------
--\n");
                    float time_ACPFPGA = 0;
                    time_ACPFPGA = (float)1.0 * (time_ACP_end -
time_ACP_start) / (COUNTS_PER_SECOND/1000000);
                    printf("Execution Time for ACP FPGA in Micro-Seconds :
%f\n" , time_ACPFPGA);
              int j;

              for (j = 0 ; j < 10 ; j++)
               {
                  printf("Input : %d , Output : %d\n " ,(int)Find_input[j],
(int)Find_output[j]);
               }
     return 0;
}

int main()
{
    init_platform();
    input();
    PS();
    Find_ACP();
    cleanup_platform();
    return 0;
}
```

JTAGTERMINAL


aeld_project_32.sdk - Debug - cw1/src/helloworld.c - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Window Help
Debug
ARM Cortex-A9 MPCore #0 (Running)
Variables Breakpoints XSCT Console Emulation Console
XSCT Process
D:\Xilinx\SDK\2019.1\bin\unw
Console Tasks
TCF Debug Virtual Term
Writable Smart Insert 139 : 1
22:34 21-03-2023

```
JTAG-based Hyperterminal.
Connected to JTAG-based Hyperterminal over TCP port : 51825
(using socket : sock636)
Help :
Terminal requirements :
  (i) Processor's STDOUT is redirected to the ARM DCC/MDM UART
  (ii) Processor's STDIN is redirected to the ARM DCC/MDM UART.
       Then, text input from this console will be sent to DCC/MDM's UART port.
  NOTE: This is a line-buffered console and you have to press "Enter"
        to send a string of characters to DCC/MDM.

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
-----------PS EXECUTION TIME--------------------
Execution Time for PS in Micro-Seconds : 1.470769

-----------ACP FPGA EXECUTION TIME--------------------
Execution Time for ACP FPGA in Micro-Seconds : 4.830769
Input : 13 , Output : 0
 Input : 3 , Output : 0
 Input : 2 , Output : 0
 Input : 9 , Output : 0
 Input : 0 , Output : 0
 Input : 8 , Output : 0
 Input : 12 , Output : 0
 Input : 16 , Output : 0
 Input : 16 , Output : 0
 Input : 19 , Output : 0
```

Q2)-



```
design_1_multi_0_1_synth_1: C:/Users/shubh/aeld/aeld_project_3/aeld_project_3.runs/design_1_multi_0_1_synth_1/runme.log
synth_1: C:/Users/shubh/aeld/aeld_project_3/aeld_project_3.runs/synth_1/runme.log
```

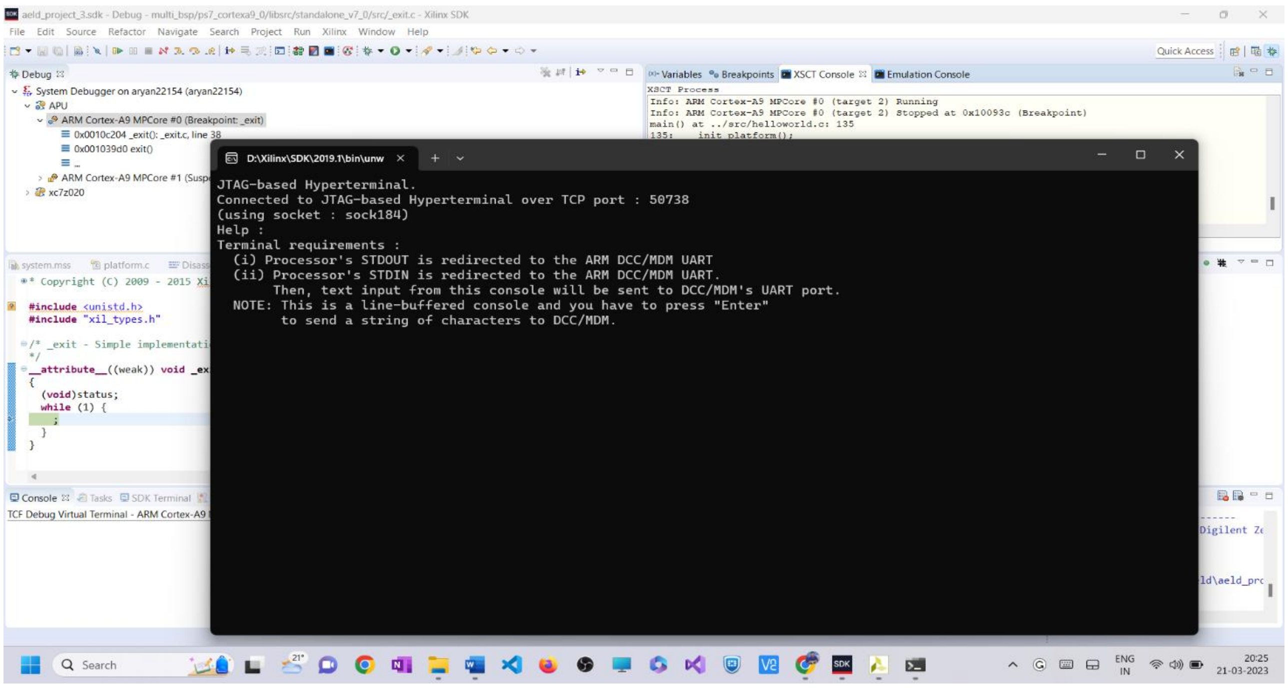
Code :-

```
/*****************************************************************************
*
*
* Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reserved.
*
* Permission is hereby granted, free of charge, to any person obtaining a copy
* of this software and associated documentation files (the "Software"), to
deal
* in the Software without restriction, including without limitation the rights
* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
* copies of the Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be included in
* all copies or substantial portions of the Software.
*
* Use of the Software is limited solely to applications:
* (a) running on a Xilinx device, or
* (b) that interact with a Xilinx device through a bus or interconnect.
```

```
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
* XILINX  BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF
* OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
* SOFTWARE.
*
* Except as contained in this notice, the name of the Xilinx shall not be used
* in advertising or otherwise to promote the sale, use or other dealings in
* this Software without prior written authorization from Xilinx.
*
******************************************************************************
/

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

#include <stdio.h>
#include <stdlib.h>
#include "xaxidma.h"
#include "xparameters.h"
#include "platform.h"
#include <xtime_l.h>
#define INP_SIZE 1024
int Find_input[INP_SIZE],Find_output[INP_SIZE];
int Find_outputps[INP_SIZE];
void input()
{
                for (int i=0;i<INP_SIZE;i++)
                {
                    Find_input[i]= (rand()%20);
                }
}
void PS()
```

```
{
    XTime time_PS_start,time_PS_end;
        XTime_SetTime(0);
        XTime_GetTime(&time_PS_start);
    for(int i=0; i<INP_SIZE; i++)
    {
        Find_outputps[i]=Find_input[i]*Find_input[i];
    }
    XTime_GetTime(&time_PS_end);
                        printf("\n-----------PS EXECUTION TIME----------------
----\n");
                            float time_processor = 0;
                            time_processor = (float)1.0 * (time_PS_end -
time_PS_start) / (COUNTS_PER_SECOND/1000000);
                            printf("Execution Time for PS in Micro-Seconds :
%f\n" , time_processor);
}
int Find_ACP()
{
    int status;

     // ACP DMA Initialization
     XAxiDma_Config *DMA_confptracp; //DMA configuration pointer
     XAxiDma AxiDMAacp; // DMA instance pointer
      // Copy the DMA information (received from hardware in xparameters.h
file)
     DMA_confptracp = XAxiDma_LookupConfig(XPAR_AXI_DMA_0_DEVICE_ID);
     status = XAxiDma_CfgInitialize(&AxiDMAacp, DMA_confptracp);
     if(status != XST_SUCCESS)
     {
        printf("ACP DMA Init Failed\t\n");
        return XST_FAILURE;
     }
                 XTime time_ACP_start , time_ACP_end;
                XTime_SetTime(0);
                XTime_GetTime(&time_ACP_start);
            // Generate random numbers

               status = XAxiDma_SimpleTransfer(&AxiDMAacp,
(UINTPTR)Find_output, (sizeof(int)*INP_SIZE),XAXIDMA_DEVICE_TO_DMA);
               status = XAxiDma_SimpleTransfer(&AxiDMAacp,
(UINTPTR)Find_input, (sizeof(int)*INP_SIZE),XAXIDMA_DMA_TO_DEVICE);
               // We have only configure the DMA to perform these two
transactions..DMA might not have started the transactions.

             //  Xil_DCacheInvalidateRange((UINTPTR)FFT_output_PLACP,
(sizeof(float complex)*FFT_Size));
               //status = checkDMAIdle(XPAR_AXI_DMA_0_BASEADDR, 0x04);
               status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x04) &
0x00000002;
               while(status!=0x00000002)
               {
                 status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x04) &
0x00000002;
               }
               status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x34) &
0x00000002;

              while(status!=0x00000002)
              {
                status = XAxiDma_ReadReg(XPAR_AXI_DMA_0_BASEADDR,0x34) &
0x00000002;
              }
              XTime_GetTime(&time_ACP_end);
                            printf("\n-----------ACP FPGA EXECUTION TIME------
--------------\n");
                                float time_ACPFPGA = 0;
                                time_ACPFPGA = (float)1.0 * (time_ACP_end -
time_ACP_start) / (COUNTS_PER_SECOND/1000000);
                                printf("Execution Time for ACP FPGA in Micro-
Seconds : %f\n" , time_ACPFPGA);
              int j;

              for (j = 0 ; j < 10 ; j++)
               {
                  printf("Input : %d , Output : %d\n " ,(int)Find_input[j],
(int)Find_output[j]);

               }

     return 0;
}

int main()
{
    init_platform();
    input();
    PS();
    Find_ACP();
    cleanup_platform();
    return 0;
}
```

JTAGTERMINAL



aeld_project_3.sdk - Debug - multi_bsp/ps7_cortexa9_0/libsrc/standalone_v7_0/src/_exit.c - Xilinx SDK

```
System Debugger on aryan22154 (aryan22154)
  APU
    ARM Cortex-A9 MPCore #0 (Breakpoint: _exit)
      0x0010c204 _exit(): _exit.c, line 38
      0x001039d0 exit()
      ...
    ARM Cortex-A9 MPCore #1 (Susp
  xc7z020
```

```
XSCT Process
Info: ARM Cortex-A9 MPCore #0 (target 2) Running
Info: ARM Cortex-A9 MPCore #0 (target 2) Stopped at 0x10093c (Breakpoint)
main() at ../src/helloworld.c: 135
135:    init_platform();
```

```
* Copyright (C) 2009 - 2015 Xi

#include <unistd.h>
#include "xil_types.h"

/* _exit - Simple implementati
 */
__attribute__((weak)) void _ex
{
  (void)status;
  while (1) {
      ;
  }
}
```

TCF Debug Virtual Terminal - ARM Cortex-A9

D:\Xilinx\SDK\2019.1\bin\unw

```
JTAG-based Hyperterminal.
Connected to JTAG-based Hyperterminal over TCP port : 50738
(using socket : sock184)
Help :
Terminal requirements :
  (i) Processor's STDOUT is redirected to the ARM DCC/MDM UART
  (ii) Processor's STDIN is redirected to the ARM DCC/MDM UART.
       Then, text input from this console will be sent to DCC/MDM's UART port.
  NOTE: This is a line-buffered console and you have to press "Enter"
        to send a string of characters to DCC/MDM.
```