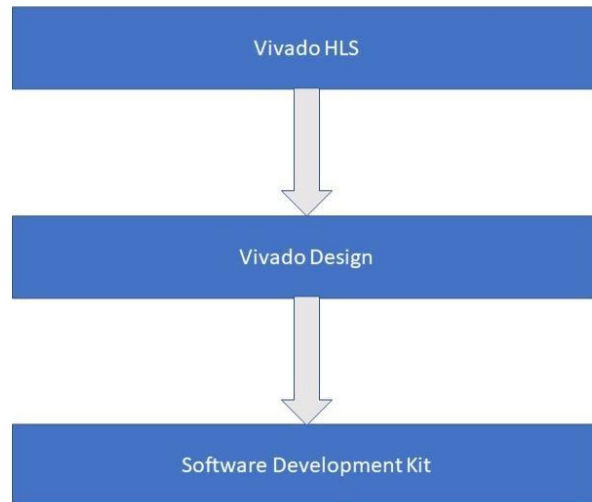


Group: Panduranga-Sood

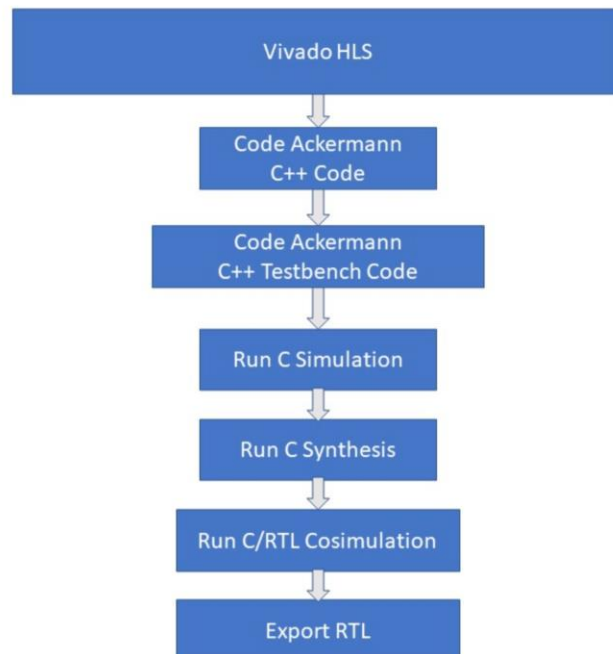
Custom Logic IP Implementation on Hardware

Custom IP Hardware Implementation Software Used Flow



Vivado HLS

Vivado HLS Implementation Steps



Vivado HLS Outputs

- C Simulation:

Vivado HLS 2019.1 - ack (D:\Ackermann_function\ack)

File Edit Project Solution Run Window Help

Debug Explorer

<terminated>ack.Debug [C/C++ Application]

<terminated, exit value: 0>gdb (8.0.1)

ack.cpp

```
16 void ack(hls::stream<int_side_ch> &inStream, hls::stream<int_side_ch> &outStream)
17 {
18     #pragma HLS INTERFACE axis port=inStream
19     #pragma HLS INTERFACE axis port=outStream
20     #pragma HLS INTERFACE s_axilite port=return bundle=CRTL_BUS
21     int_side_ch val_in;
22     int_side_ch val_out;
23     int x,y;
24     val_in= inStream.read();
25     x=(unsigned int)val_in.data;
26     val_in= inStream.read();
27     y=(unsigned int)val_in.data;
28     cout<<"X is "<<x<<endl;
29     cout<<"Y is "<<y<<endl;
30
31     int value[30000];
32     size_t size = 0;
```

Console

<terminated> (exit value: 0) ack.Debug [C/C++ Application] csim.exe

X is 3

Y is 11

Ackermann of X and Y is = 16381

- C Synthesis:

Timing (ns)

Summary

| Clock | Target | Estimated | Uncertainty |
|--------|--------|-----------|-------------|
| ap_clk | 10.00 | 6.774 | 1.25 |

Latency (clock cycles)

Utilization Estimates

Summary

| Name | BRAM_18K | DSP48E | FF | LUT | URAM |
|-----------------|----------|--------|-------|-------|------|
| DSP | - | - | - | - | - |
| Expression | - | - | 0 | 356 | - |
| FIFO | - | - | - | - | - |
| Instance | 0 | - | 36 | 40 | - |
| Memory | 64 | - | 0 | 0 | 0 |
| Multiplexer | - | - | - | 270 | - |
| Register | - | - | 290 | - | - |
| Total | 64 | 0 | 326 | 666 | 0 |
| Available | 100 | 66 | 28800 | 14400 | 0 |
| Utilization (%) | 64 | 0 | 1 | 4 | 0 |

- C/RTL Cosimulation:

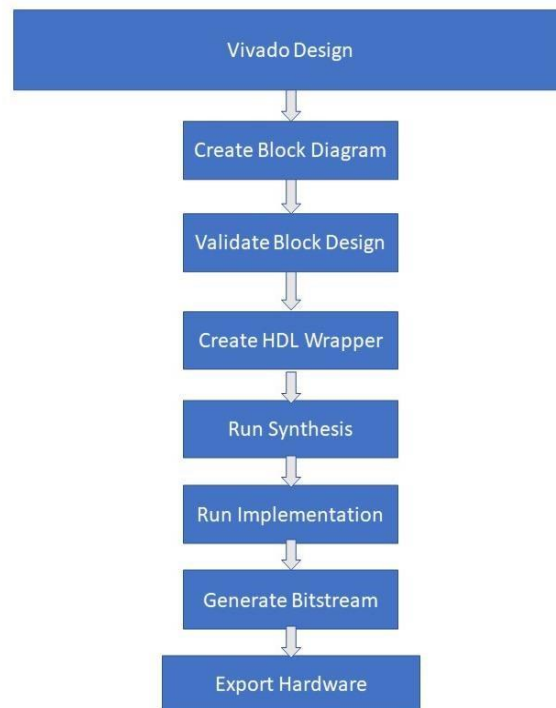
Cosimulation Report for 'ack'

| Result | | | | | | | |
|---------|--------|----------|----------|----------|----------|-----|-----|
| RTL | Status | Latency | | | Interval | | |
| | | min | avg | max | min | avg | max |
| VHDL | NA | NA | NA | NA | NA | NA | NA |
| Verilog | Pass | 89396668 | 89396668 | 89396668 | NA | NA | NA |

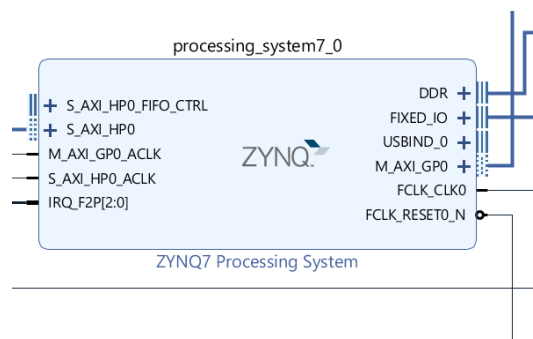
Export the report(.html) using the [Export Wizard](#)

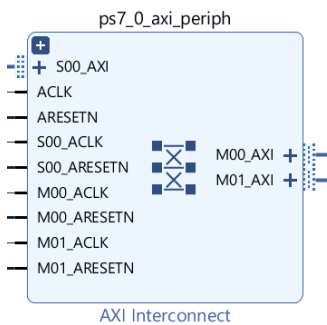
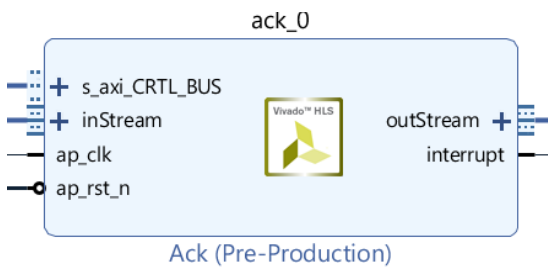
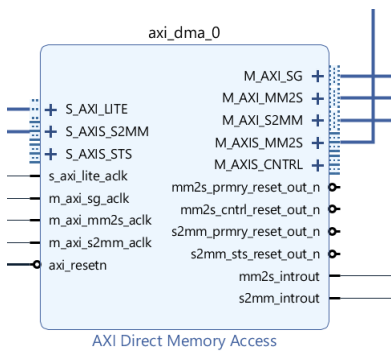
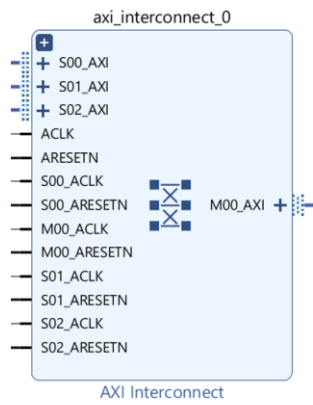
Vivado Design

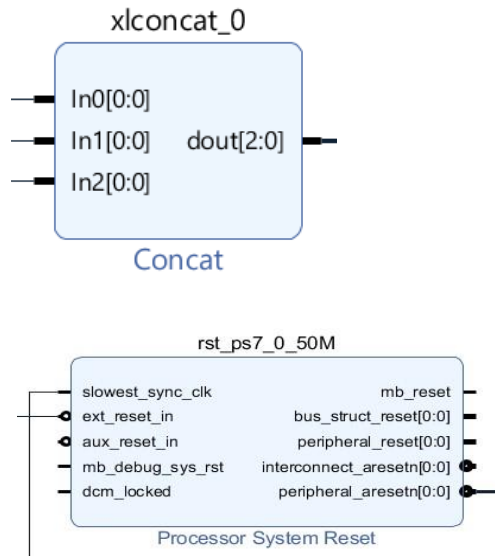
Vivado Design Implementation Steps



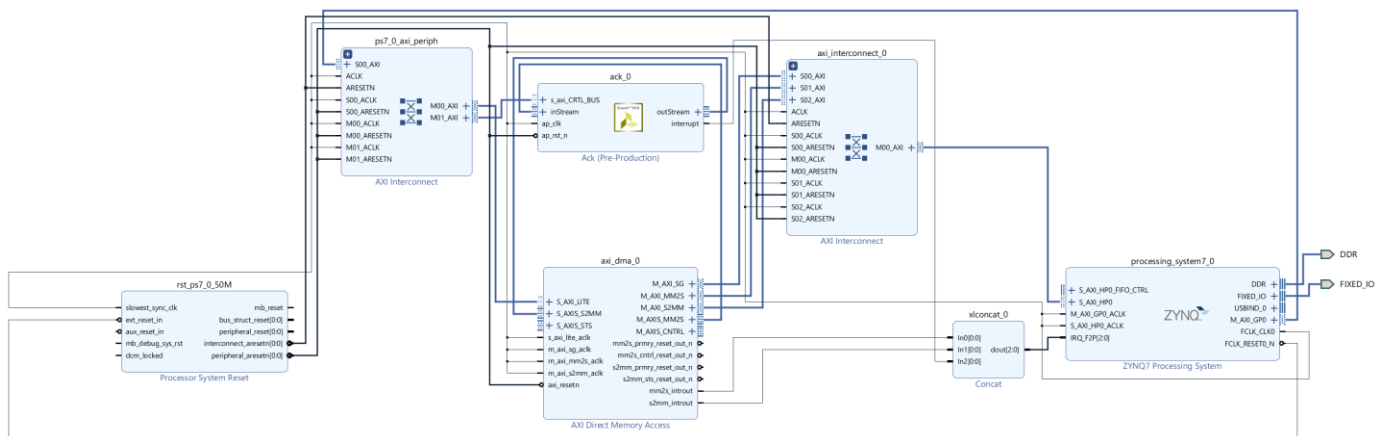
Vivado Design design blocks



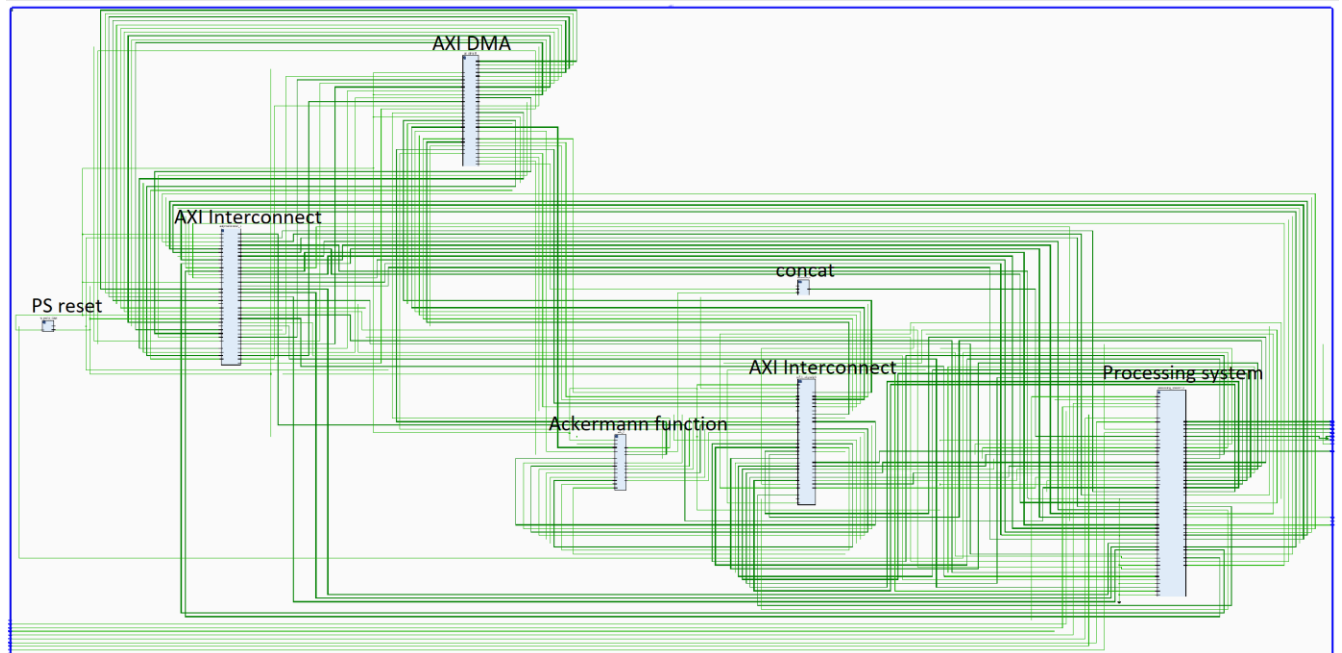




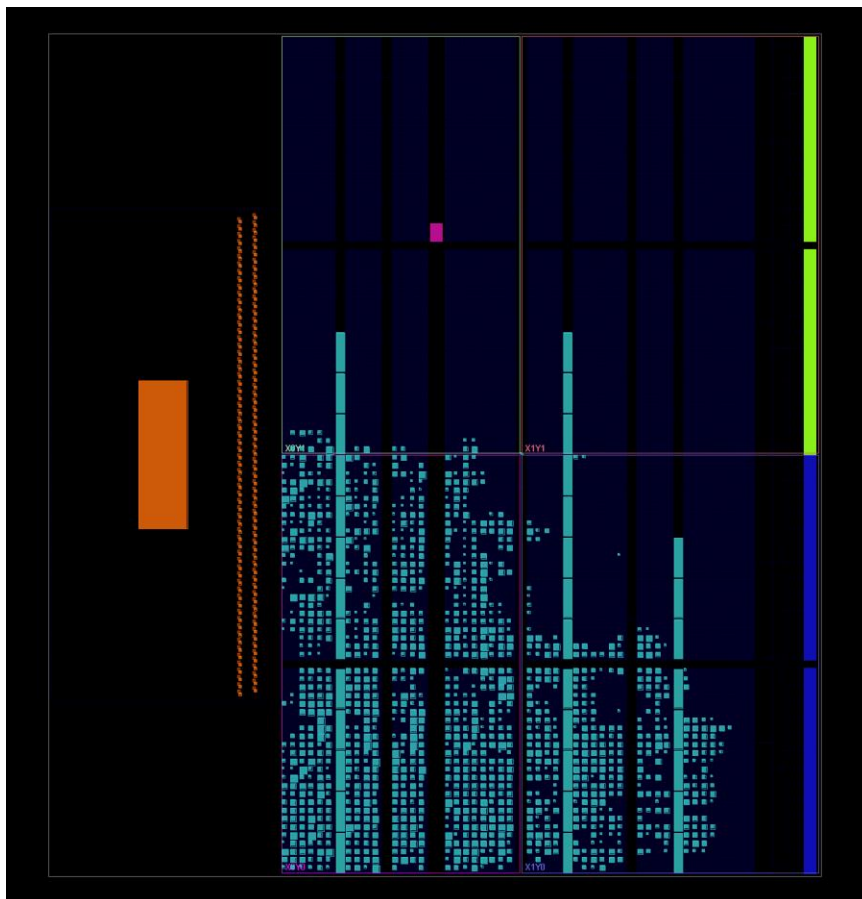
Block Design



Schematic Design



LUTs used in the Design



Vivado Design Outputs

- Timing Report:

Design Timing Summary

| Setup | Hold | Pulse Width |
|--------------------------------------|----------------------------------|---|
| Worst Negative Slack (WNS): 8.153 ns | Worst Hold Slack (WHS): 0.010 ns | Worst Pulse Width Slack (WPWS): 8.750 ns |
| Total Negative Slack (TNS): 0.000 ns | Total Hold Slack (THS): 0.000 ns | Total Pulse Width Negative Slack (TPWS): 0.000 ns |
| Number of Failing Endpoints: 0 | Number of Failing Endpoints: 0 | Number of Failing Endpoints: 0 |
| Total Number of Endpoints: 11096 | Total Number of Endpoints: 11096 | Total Number of Endpoints: 4281 |

All user specified timing constraints are met.

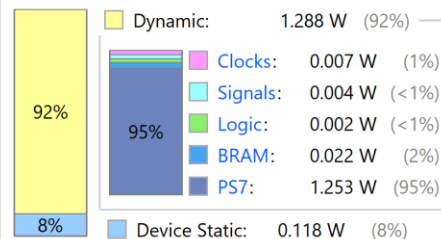
- Power Summary:

Summary

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

Total On-Chip Power: 1.407 W
Design Power Budget: Not Specified
Power Budget Margin: N/A
Junction Temperature: 41.2°C
Thermal Margin: 43.8°C (3.7 W)
Effective θ_{JA} : 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

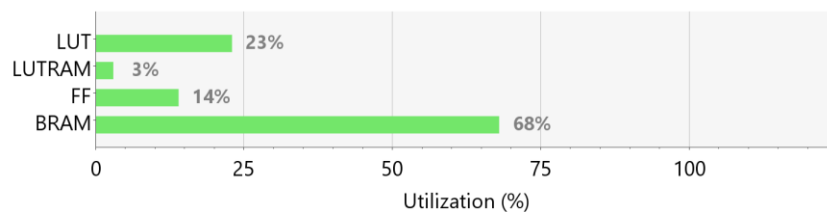
On-Chip Power



- Utilization Report:

Summary

| Resource | Utilization | Available | Utilization % |
|----------|-------------|-----------|---------------|
| LUT | 3327 | 14400 | 23.10 |
| LUTRAM | 206 | 6000 | 3.43 |
| FF | 3948 | 28800 | 13.71 |
| BRAM | 34 | 50 | 68.00 |



Software Development Kit

Software Development Kit Implementation Steps



Software Development Kit Output

The screenshot displays the Xilinx SDK interface. The 'Project Explorer' on the left shows the project structure, including the 'ack' directory, 'src' directory with 'helloworld.c', and 'ack_wrapper_hw_platform_0' directory. The 'helloworld.c' file is open in the editor, showing the Ackermann function implementation. The 'SDK Terminal' at the bottom shows the output of the program, including the initialization of the Ackermann function, the input values (M=3, N=12), and the resulting output (32765). The 'SDK Log' on the right shows the system messages and the command used to launch the application.

```
ack.sdk - C/C++ - ack/src/helloworld.c - Xilinx SDK
File Edit Navigate Search Project Run Xilinx Window Help

Project Explorer
  ack
  > Binaries
  > Includes
  > Debug
  > src
    > helloworld.c
    > platform_config.h
    > platform.c
    > platform.h
    > lscript.ld
    > Xilinx.spec
  > ack_bsp
  > ack_wrapper_hw_platform_0
    > drivers
      > ack_wrapper.bit
      > ps7_init_gpl.c
      > ps7_init_gpl.h
      > ps7_init.c
      > ps7_init.h
      > ps7_init.html
      > ps7_init.tcl
      > system.hdf

Target Connections
  > Hardware Server
  > Linux TCF Agent
  > QEMU TcfGdbClient

system.hdf
system.mss
helloworld.c

XAXiDma_IntrDisable(&axidma, XAXIDMA_IRQ_ALL_MASK, XAXIDMA_DEVICE_TO_DMA);
XAXiDma_IntrDisable(&axidma, XAXIDMA_IRQ_ALL_MASK, XAXIDMA_DMA_TO_DEVICE);
//XAXiDma_IntrEnable(&axidma, XAXIDMA_IRQ_ALL_MASK, XAXIDMA_DEVICE_TO_DMA);
//XAXiDma_IntrEnable(&axidma, XAXIDMA_IRQ_ALL_MASK, XAXIDMA_DMA_TO_DEVICE);
}

int main() {
  init_platform();
  int *m_dma_buffer_Rx = (int*) RX_BUFFER_BASE;
  print("Hello World\n\r");
  initperipherals();

  instreamdata[0] = 3;
  instreamdata[1] = 12;

  Problems Tasks Console Properties SDK Terminal SDK Log

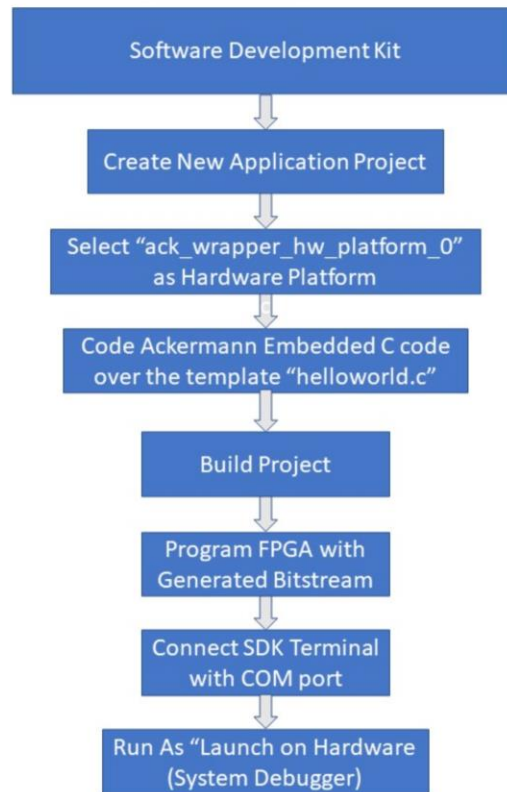
Connected to: Serial ( COM8, 115200, 0, 8 )

initializing Ackermann Function...
initializing AXI DMA core
The input M is 3
The input N is 12
sending data to axi_dma...
read data ...
calculation complete.
The Ackermann Output is: 32765
Counts per second= 325000000
Clock cycles=9304459643
28629106.59

11:47:08 INFO : Context for processor 'ps7_cortexa9_0' is
11:47:08 INFO : The application 'D:/Ackermann_function/ack
11:47:08 INFO : 'configparams force-mem-access 0' command
11:47:08 INFO : -----XSDS Script-----
connect -url tcp:127.0.0.1:3121
source D:/Ackermann_function/ack/ack.sdk/ack_wrapper_hw_plat
targets -set -nocase -filter {name =~ "APU*" && jtag_cable_na
loadhw -hw D:/Ackermann_function/ack/ack.sdk/ack_wrapper_hw_
configparams force-mem-access 1
targets -set -nocase -filter {name =~ "APU*" && jtag_cable_na
stop
ps7_init
ps7_post_config
targets -set -nocase -filter {name =~ "ARM*#0" && jtag_cable
rst -processor
targets -set -nocase -filter {name =~ "ARM*#0" && jtag_cable
dow D:/Ackermann_function/ack/ack.sdk/ack/Debug/ack.elf
```


Custom Logic Implementation on Software

Custom Logic Software Implementation Flow



Software Development Kit Output

The screenshot shows the Xilinx SDK IDE with the 'helloworld.c' file open. The code is as follows:

```
system.hdf system.mss helloworld.c
XTime start,stop;
XTime_GetTime(&start);

int m = 3, n = 10;
printf("Hello World\n");
printf("Input M is %d\n",m);
printf("Input N is %d\n",n);
printf("Ackermann of M and N is %d\n", ack_da(m, n));

XTime_GetTime(&stop);
printf("Counts per second= %d \nClock cycles=%llu \n%.2f ",COUNTS_PER_SECOND,(stop-
```

The SDK Terminal shows the output of the program:

```
Connected to: Serial ( COM8, 115200, 0, 8 )

Connected to COM8 at 115200
Hello World
Input M is 3
Input N is 10
Ackermann of M and N is 8189
Counts per second= 325000000
Clock cycles=592339039
1822581.66
```

The SDK Log shows the following messages:

```
12:17:23 INFO : Memory regions updated for context APU
12:17:23 INFO : Context for processor 'ps7_cortexa9_0' is
12:17:23 INFO : 'con' command is executed.
12:17:23 INFO : -----XSD8 Script (After Launch
targets -set -nocase -filter {name =~ "ARM*#0" && jtag_cabl
con
-----End of Script-----
12:17:23 INFO : Launch script is exported to file 'D:\sd
```

Hardware vs Software Performance Comparison

Clock cycles taken by Hardware: 9304459643

Clock cycles taken by Software: 592339039

Zybo Board Clock Period: 10ns

Therefore,

Time taken by Hardware Implementation: $9304459643 * 10\text{ns} = 93044.59643 \text{ ms}$

Time taken by Software Implementation: $592339039 * 10\text{ns} = 5923.39039 \text{ ms}$

Hence,

Hardware is slower than Software by a factor of ~83 for the Ackermann logic implementation