ELD Lab 7
Exploring ARM Processor

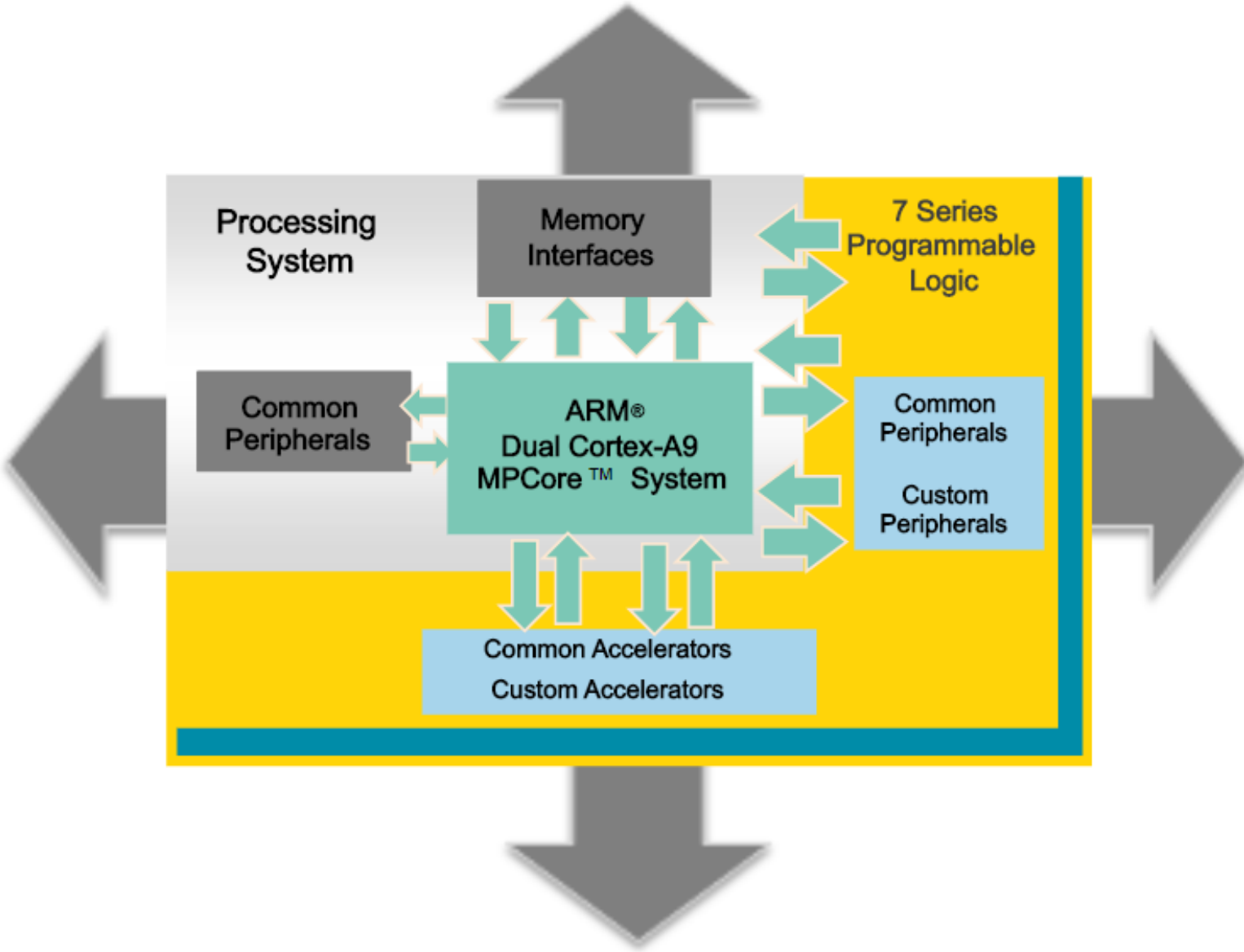
Objective

- Run simple programs on ARM Cortex A9 processor of Zynq SoC
- Homework: Implement 2x2 matrix multiplication on ARM Processor

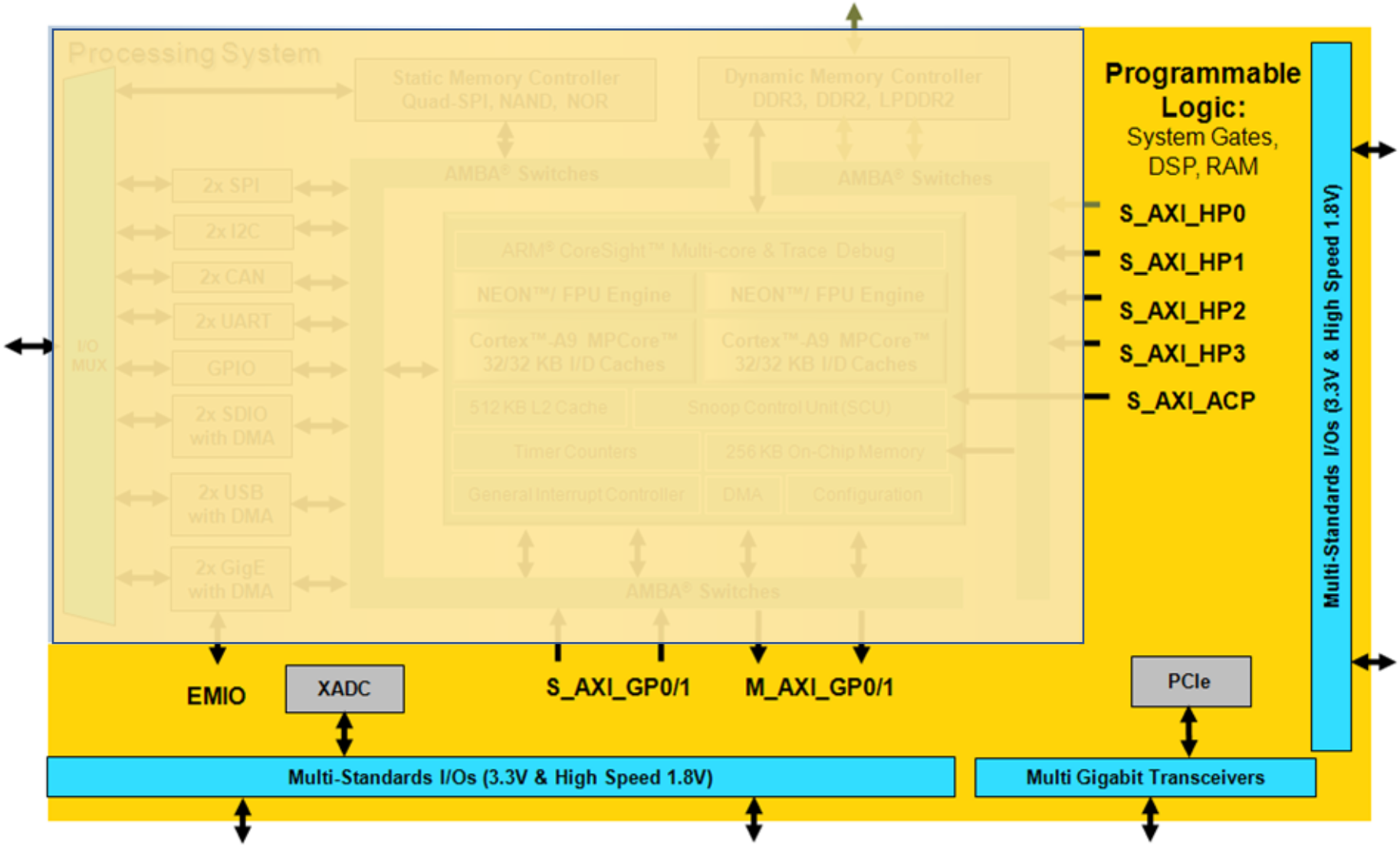
Theory

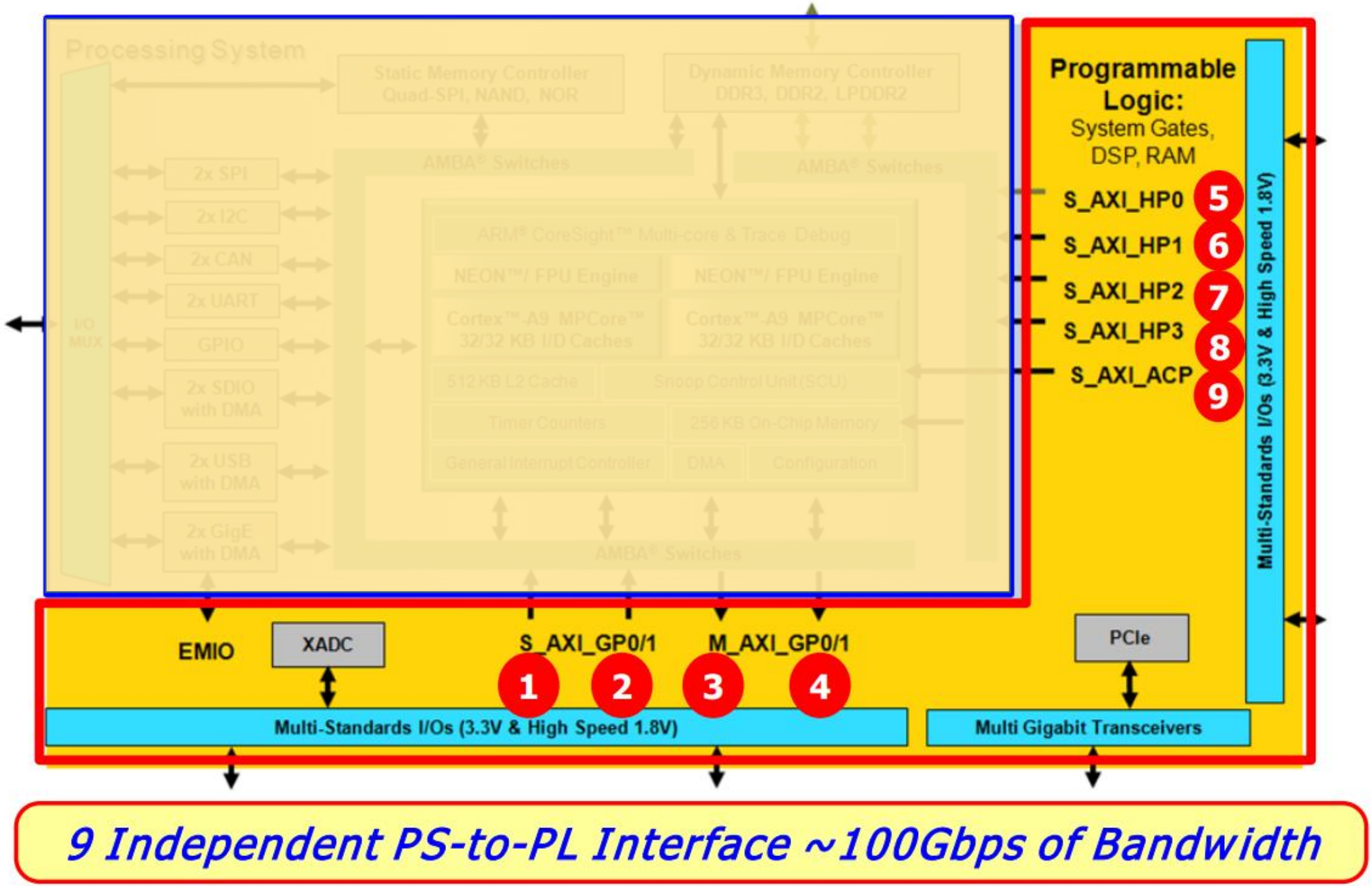
Zynq SoC

Zynq Architecture: PS and PL



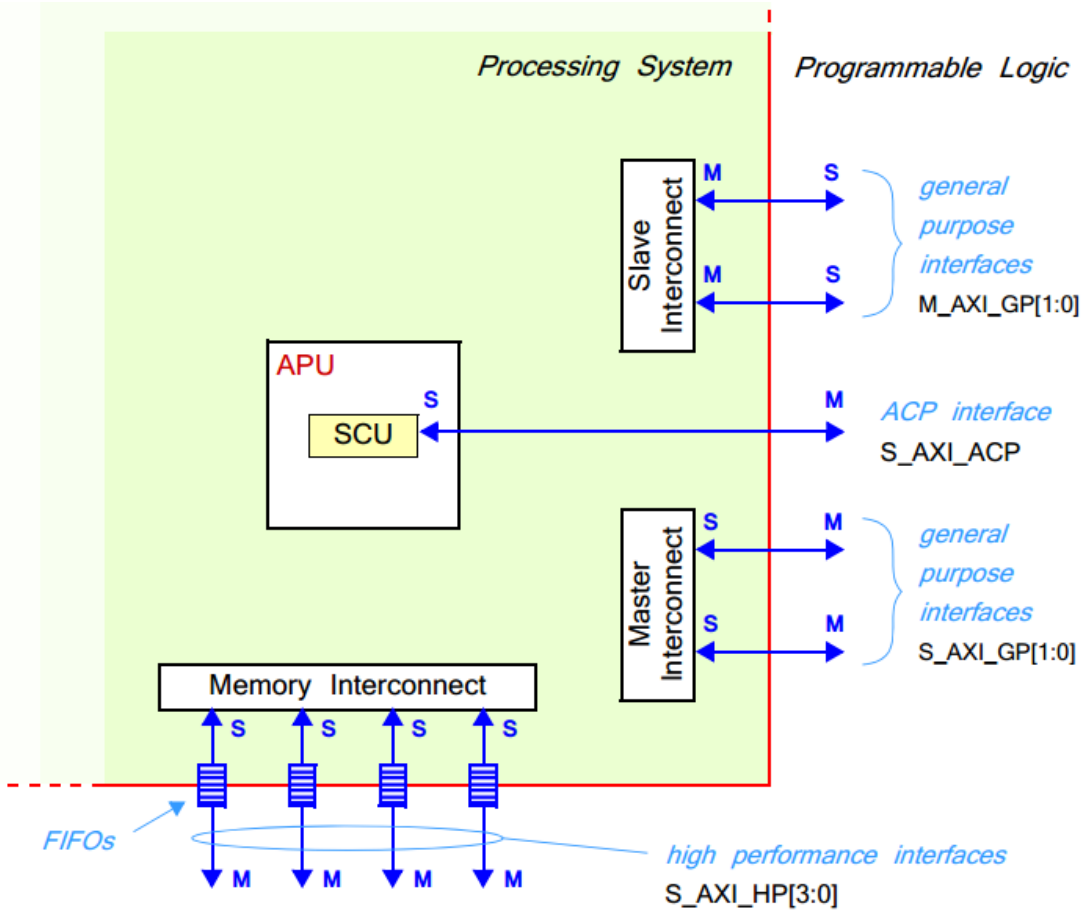
Zynq Architecture: PS and PL



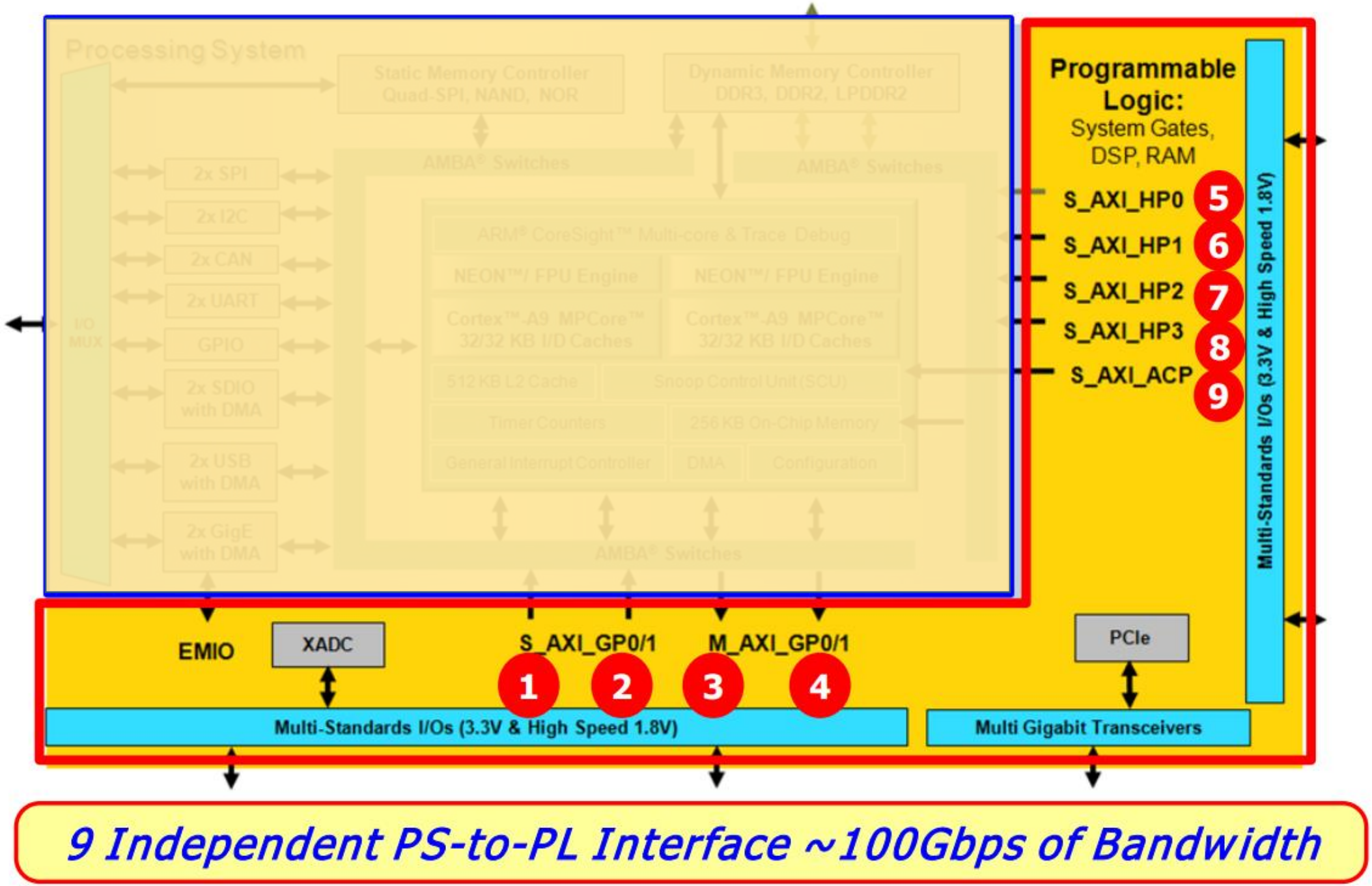


9 Independent PS-to-PL Interface ~100Gbps of Bandwidth

Zynq PS-PL Interface

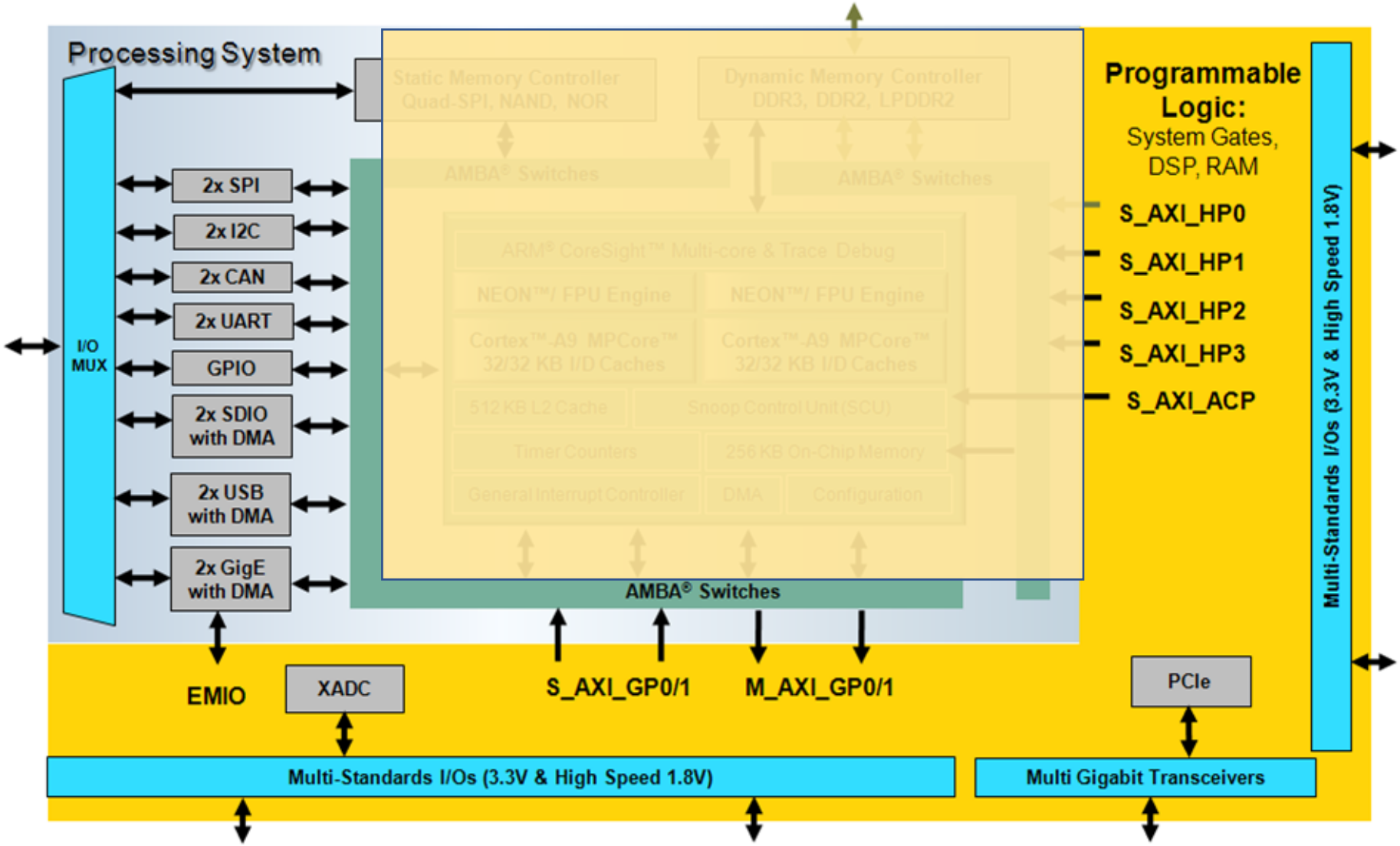


Interface Name	Interface Description	Master	Slave
M_AXI_GP0	General Purpose (AXI_GP)	PS	PL
M_AXI_GP1		PS	PL
S_AXI_GP0	General Purpose (AXI_GP)	PL	PS
S_AXI_GP1		PL	PS
S_AXI_ACP	Accelerator Coherency Port, cache-coherent transaction (ACP)	PL	PS
S_AXI_HP0	High Performance ports (AXI_HP) with read/write FIFOs and two dedicated memory ports on DDR controller and a path to the OCM. The AXI_HP interfaces are known also as AFI.	PL	PS
S_AXI_HP1		PL	PS
S_AXI_HP2		PL	PS
S_AXI_HP3		PL	PS

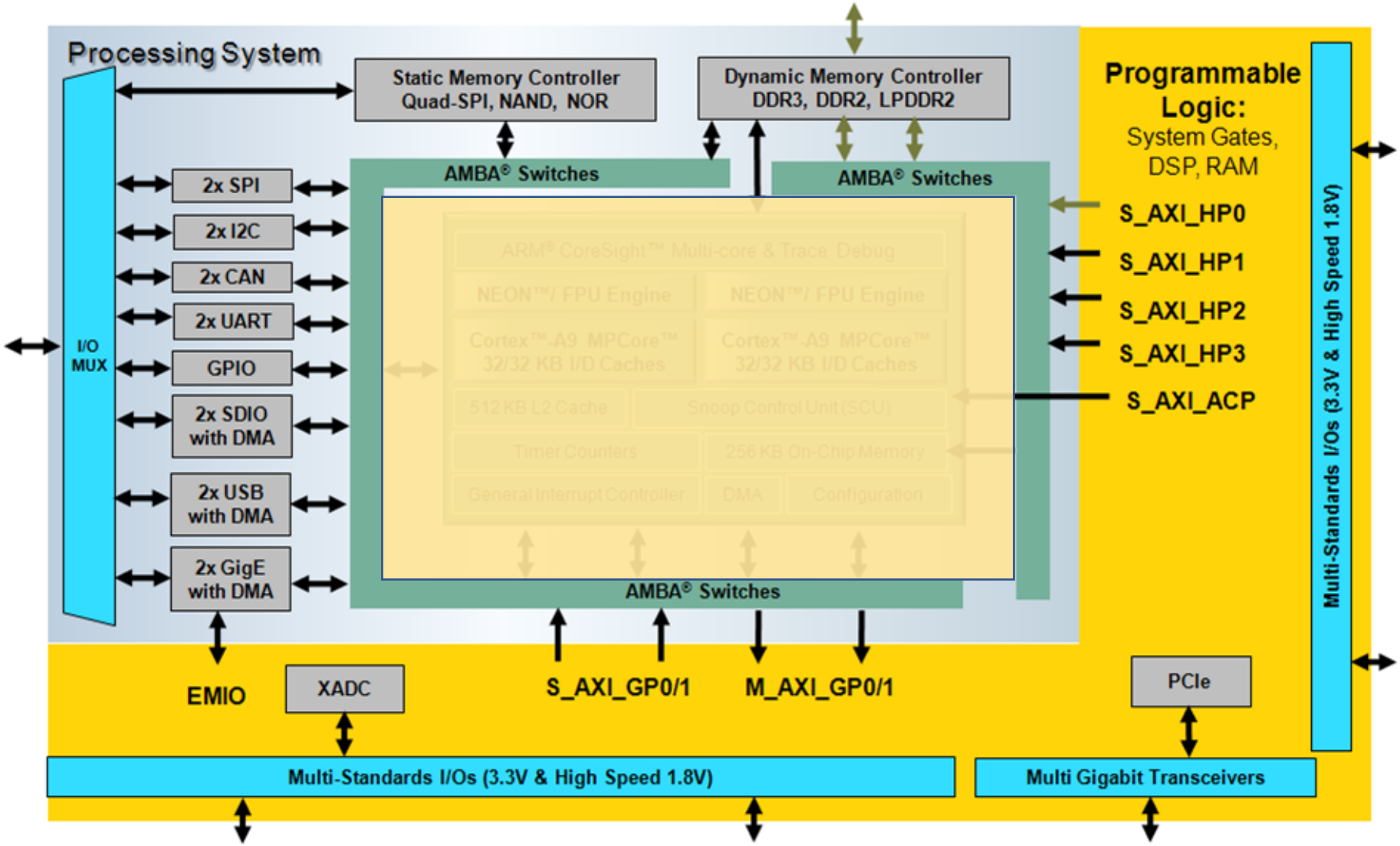


9 Independent PS-to-PL Interface ~100Gbps of Bandwidth

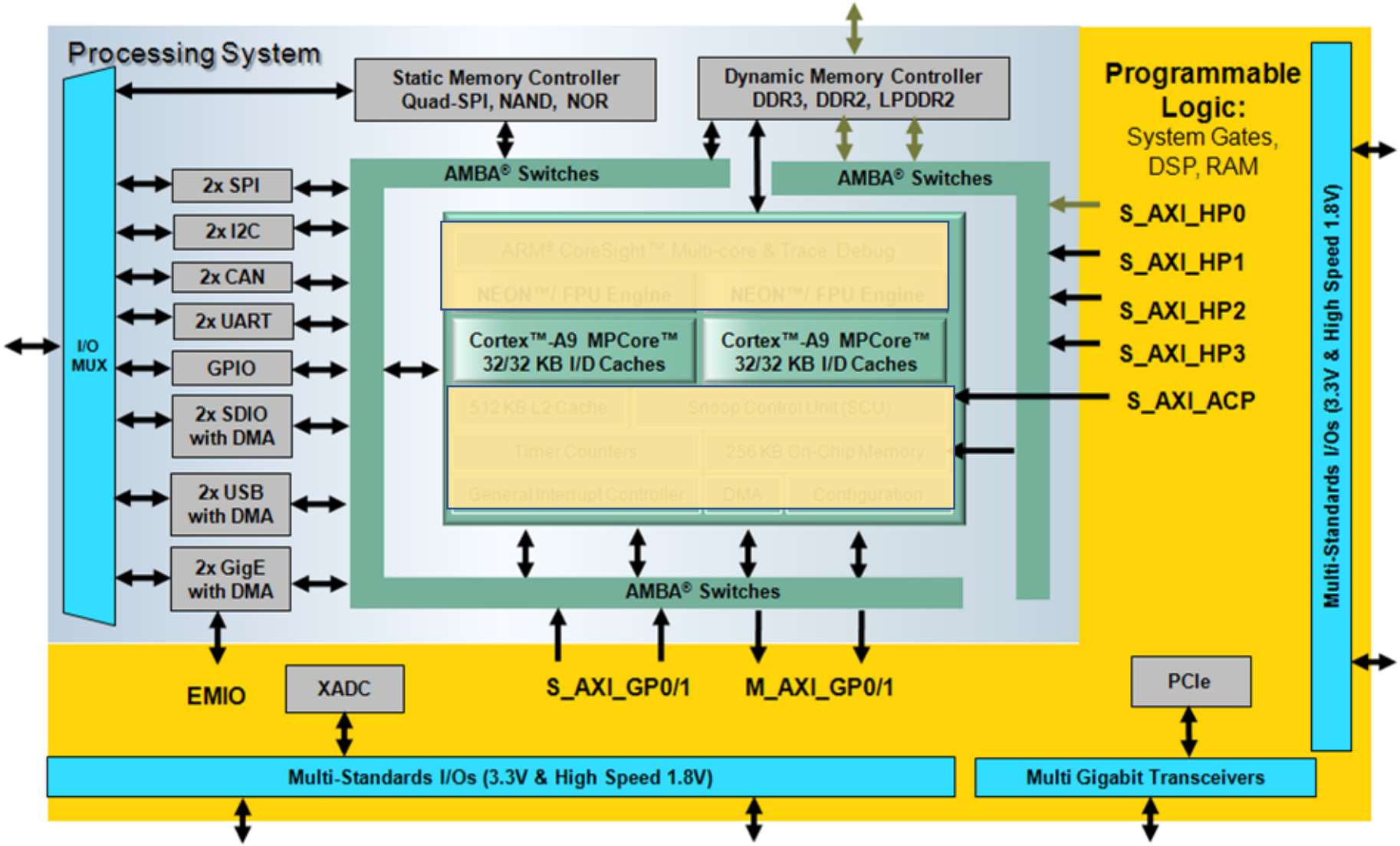
Zynq Architecture: PS and PL



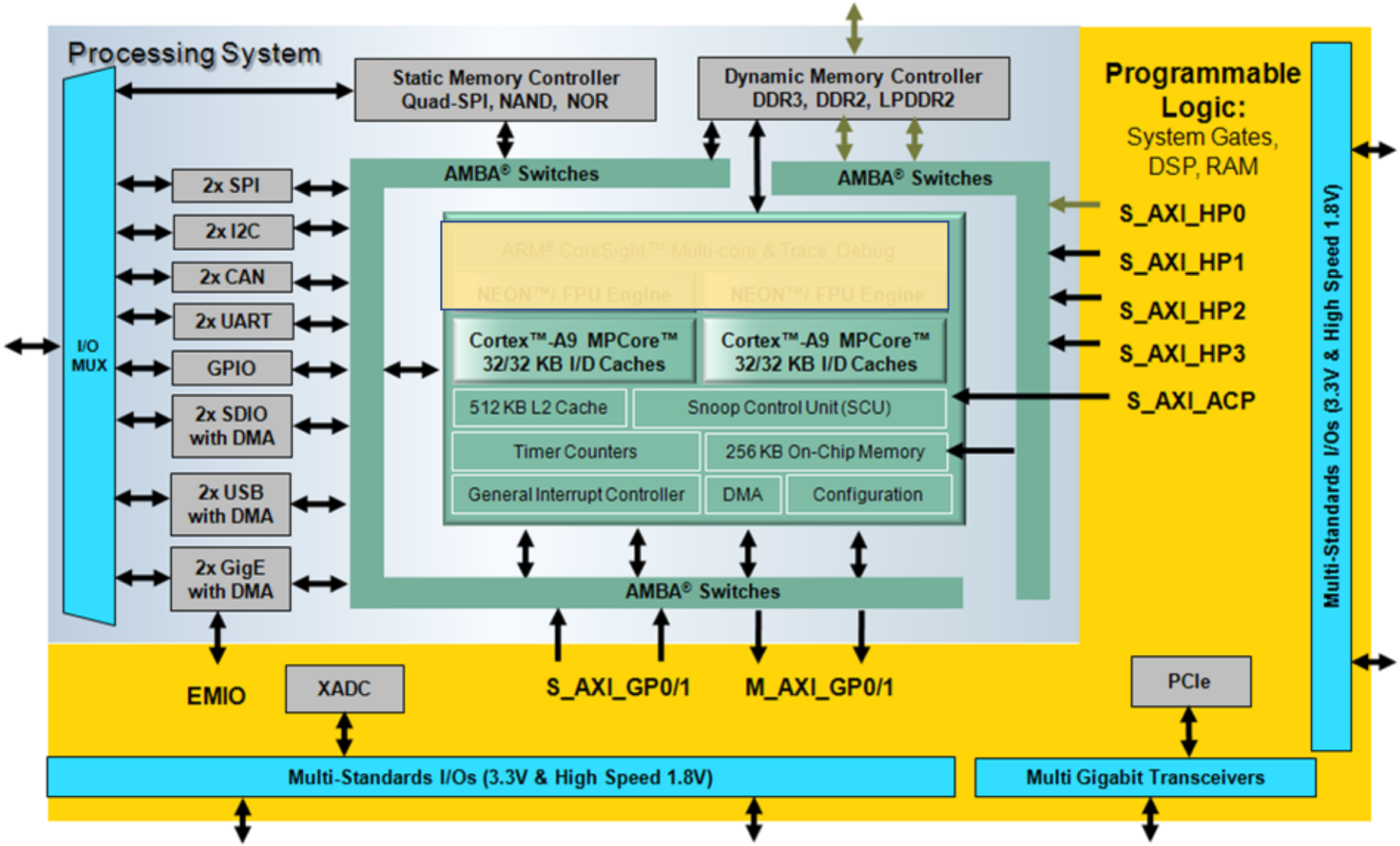
Zynq Architecture: PS and PL



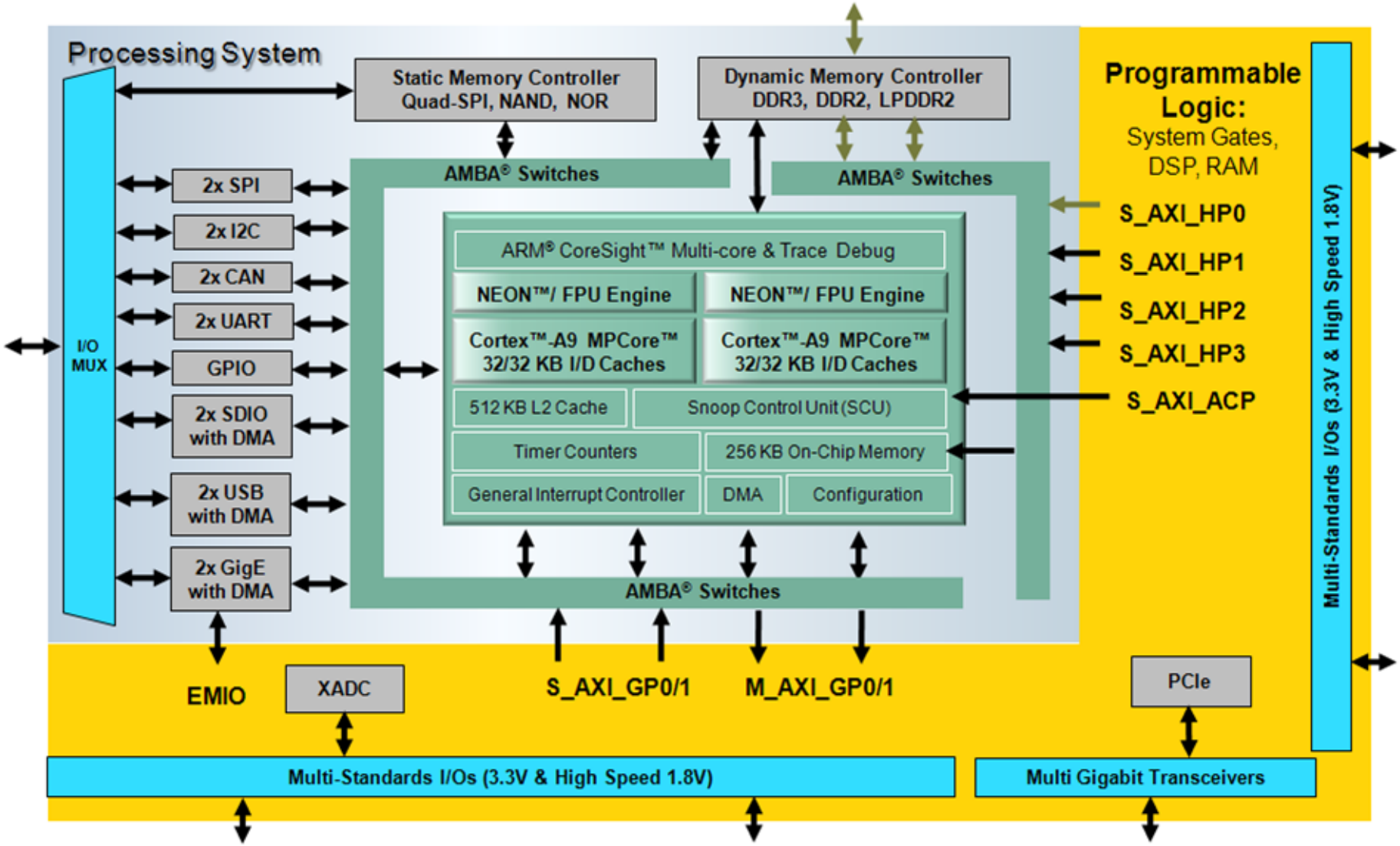
Zynq Architecture: PS and PL

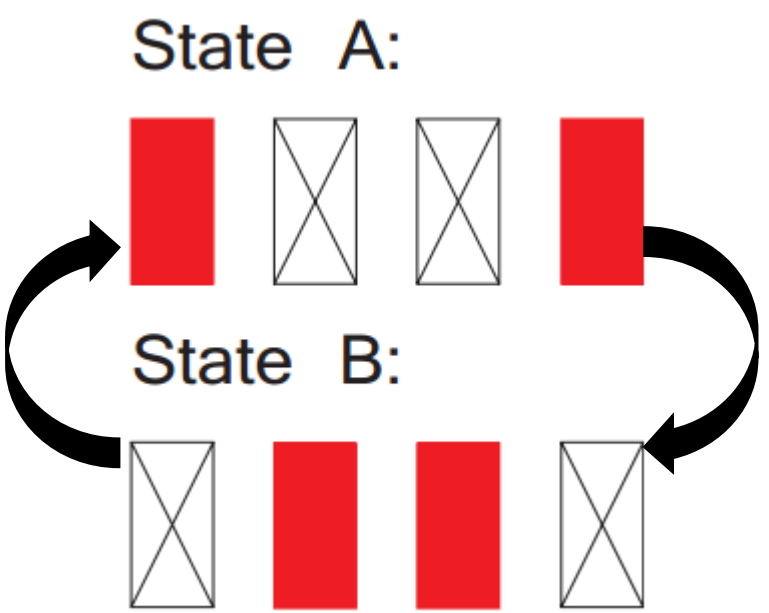


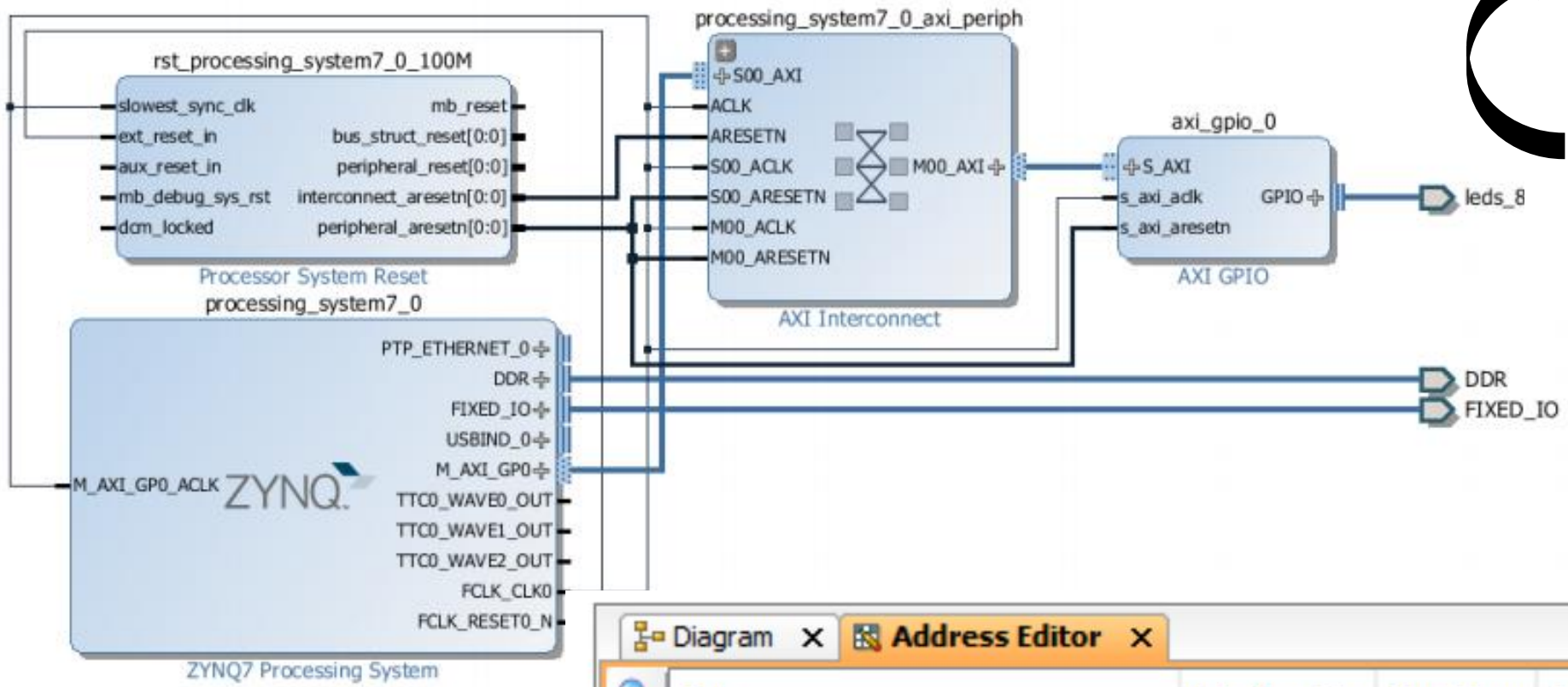
Zynq Architecture: PS and PL

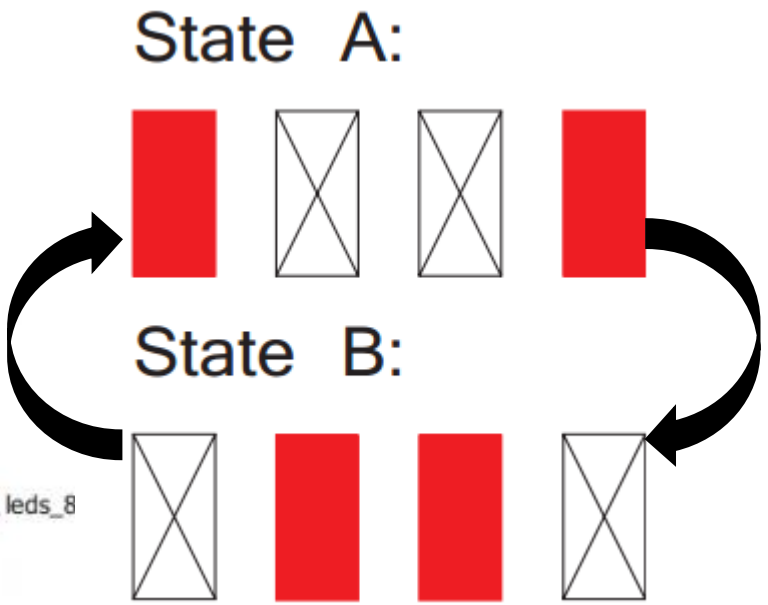


Zynq Architecture: PS and PL




State A:
State B:


rst_processing_system7_0_100M
slowest_sync_clk
ext_reset_in
aux_reset_in
mb_debug_sys_rst
dcm_locked
mb_reset
bus_struct_reset[0:0]
peripheral_reset[0:0]
interconnect_aresetn[0:0]
peripheral_aresetn[0:0]
Processor System Reset
processing_system7_0_axi_periph
S00_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
M00_AXI
AXI Interconnect
axi_gpio_0
S_AXI
s_axi_aclk
s_axi_aresetn
GPIO
AXI GPIO
leds_8
processing_system7_0
PTP_ETHERNET_0
DDR
FIXED_IO
USBIND_0
M_AXI_GP0
M_AXI_GP0_ACLK
TTC0_WAVE0_OUT
TTC0_WAVE1_OUT
TTC0_WAVE2_OUT
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
DDR
FIXED_IO


State A:
State B:

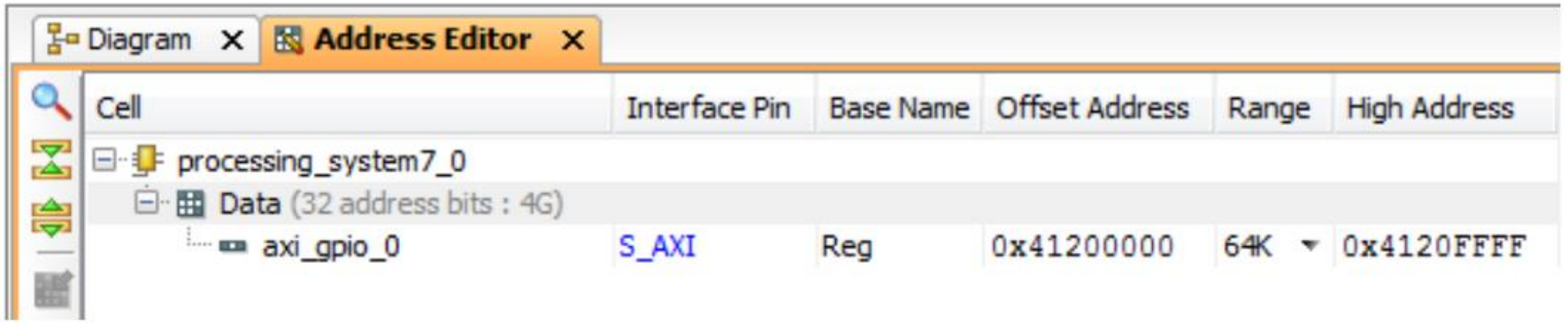
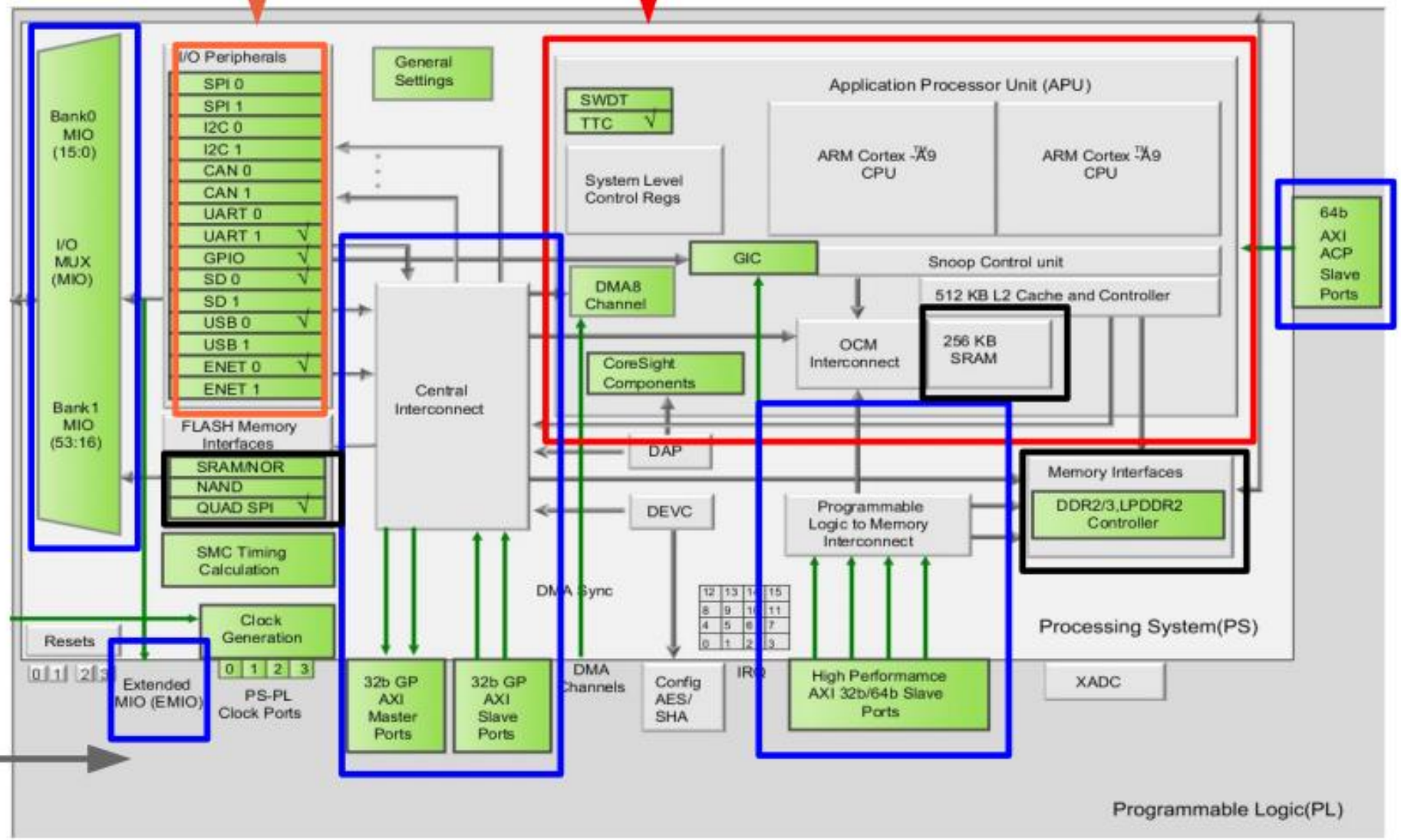


Diagram
Address Editor
Cell
Interface Pin
Base Name
Offset Address
Range
High Address
processing_system7_0
Data (32 address bits : 4G)
axi_gpio_0
S_AXI
Reg
0x41200000
64K
0x4120FFFF

Interconnections
AXI
Peripherals
Memory
User defined

Memory
On-chip
DDR
Flash

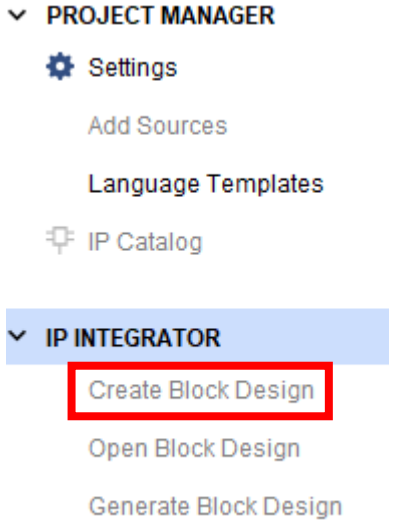
Programmable Logic

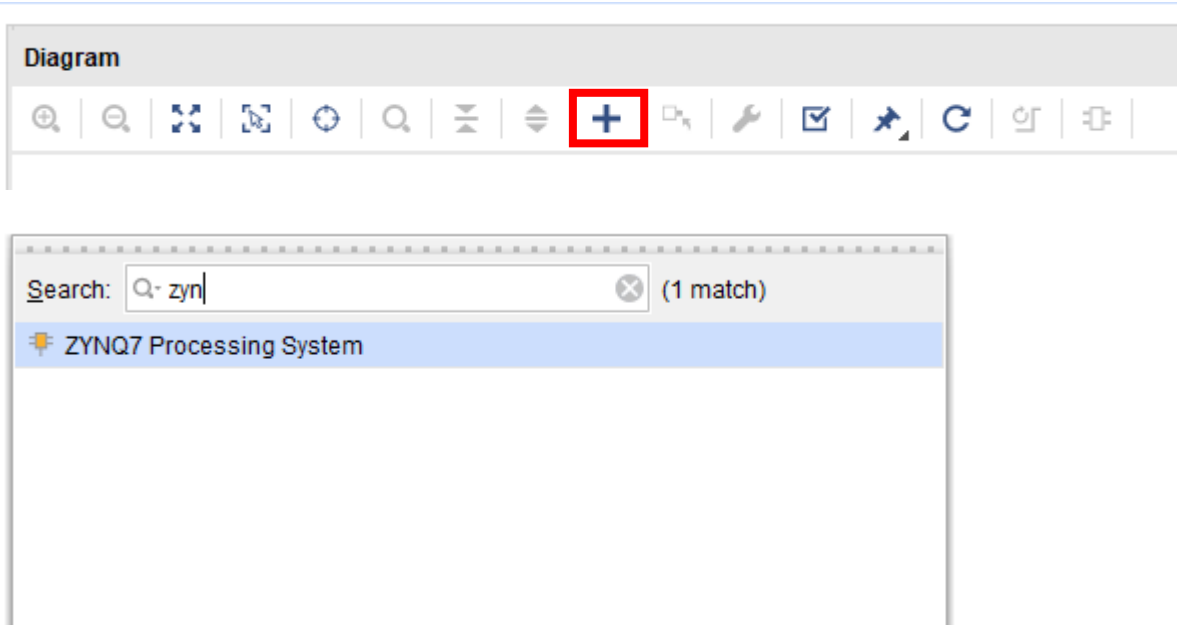


Lab

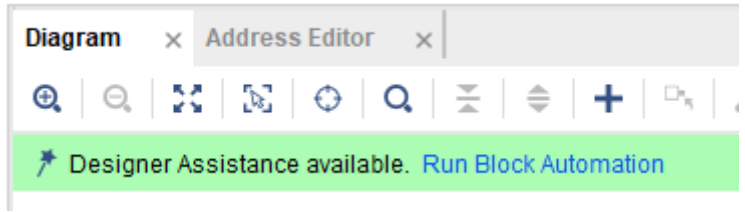
Vivado: Empty Project

- Use either Zedboard or Zybo
- Create block diagram and add Zynq IP

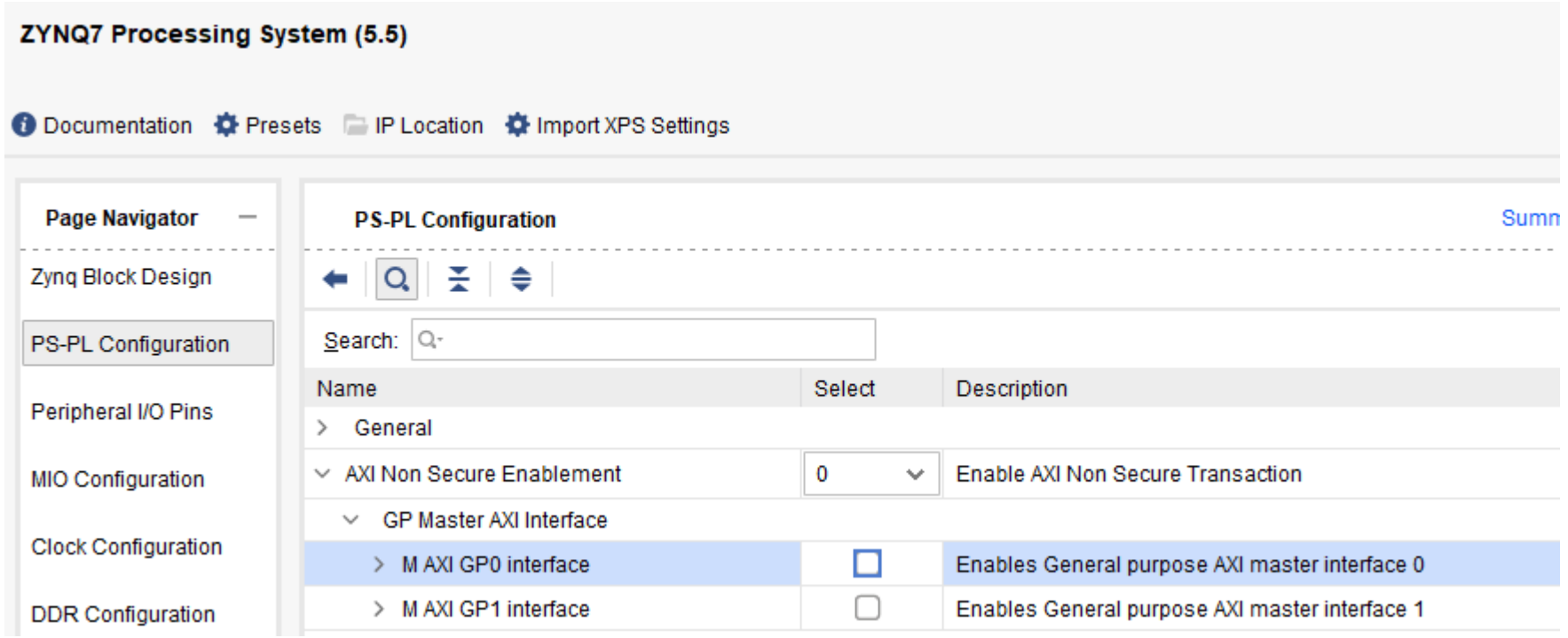




Zynq IP Configuration

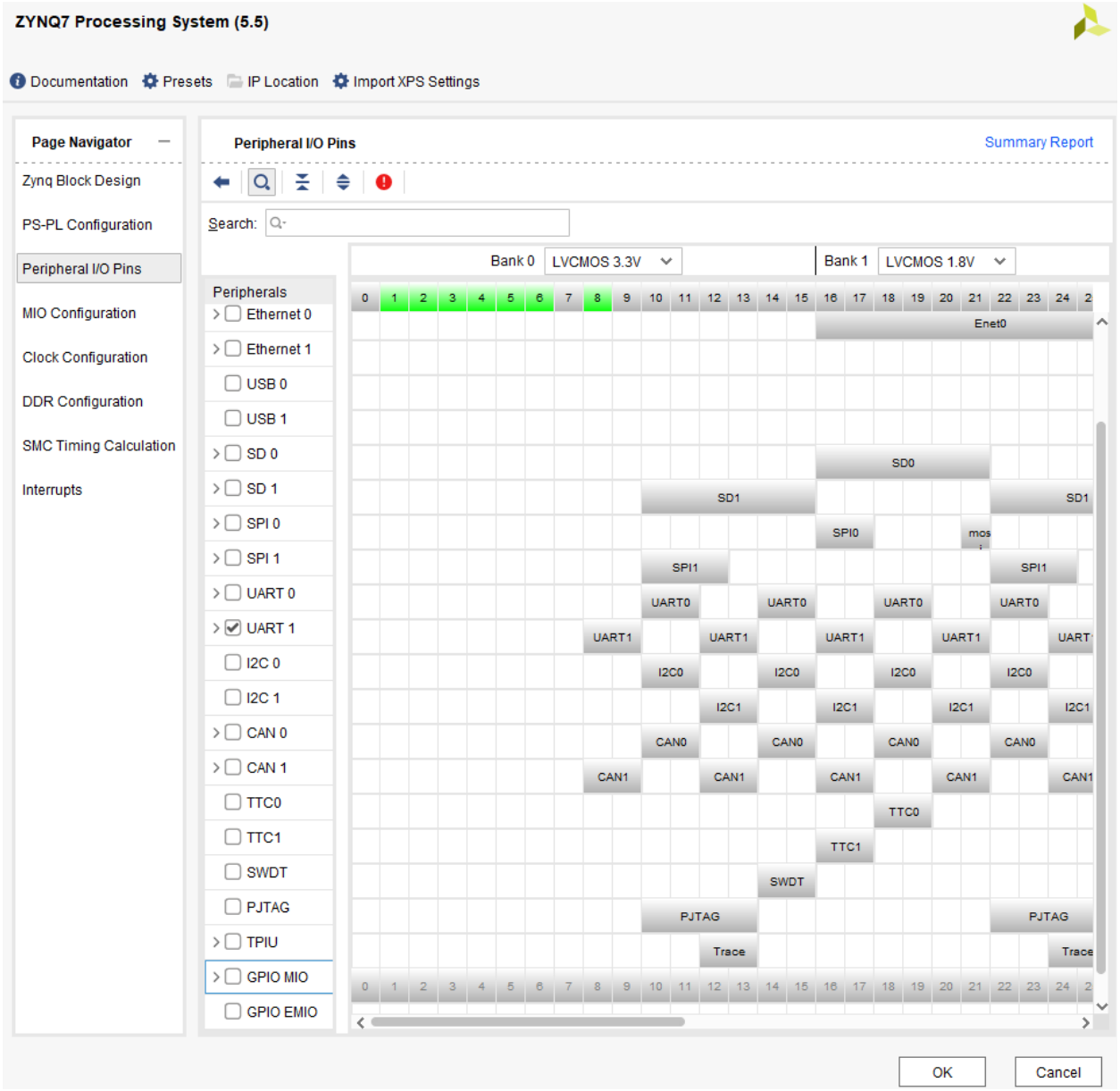


- Click on Run Block Automation and Select OK
- Remove following ports by configuring the Zynq IP: GP Master Interface



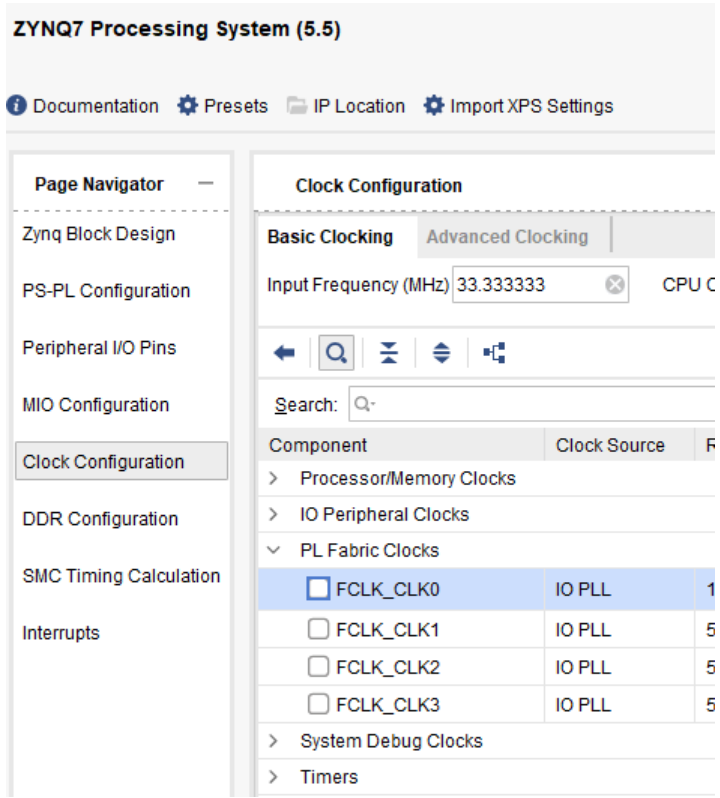
Zynq IP Configuration

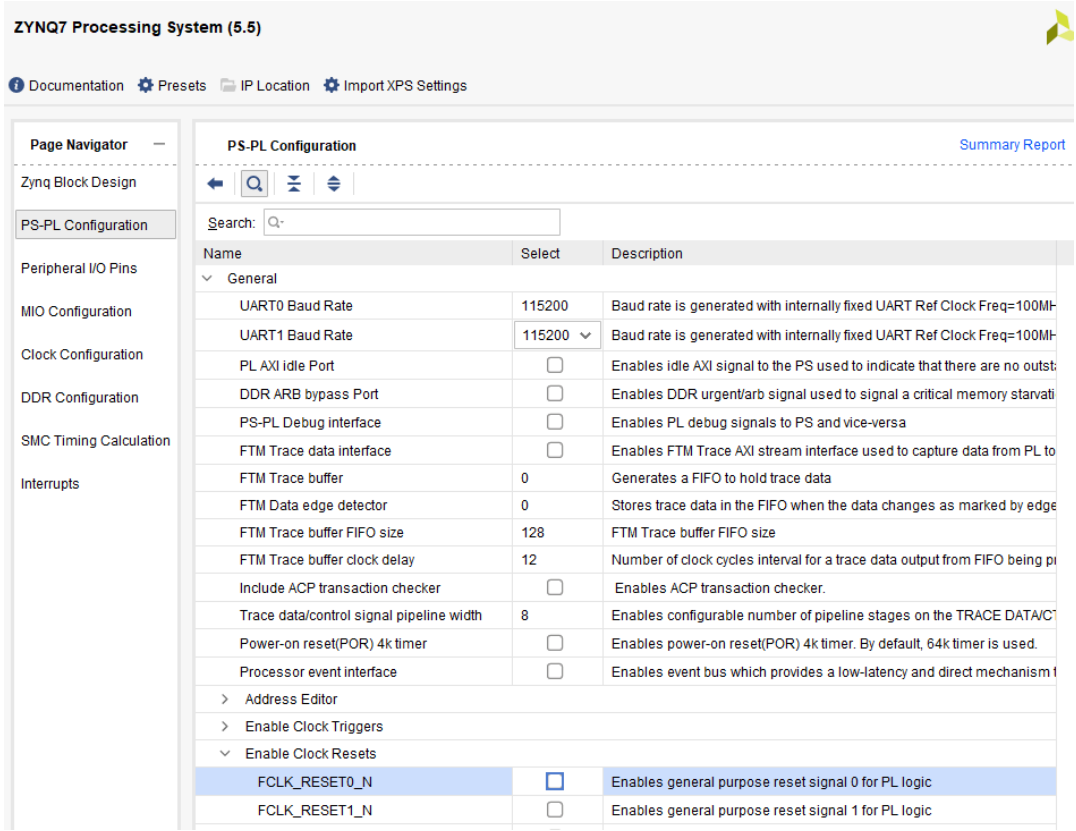
- Remove following ports by configuring the Zynq IP: Ethernet0, USB0, SD0, TTC0, GPIO MIO

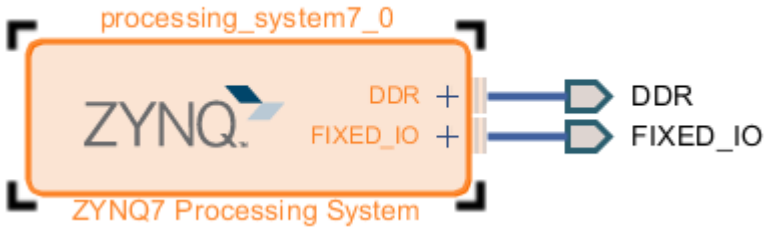


Zynq IP Configuration

- Remove following ports by configuring the Zynq IP: Fabric Clock and reset





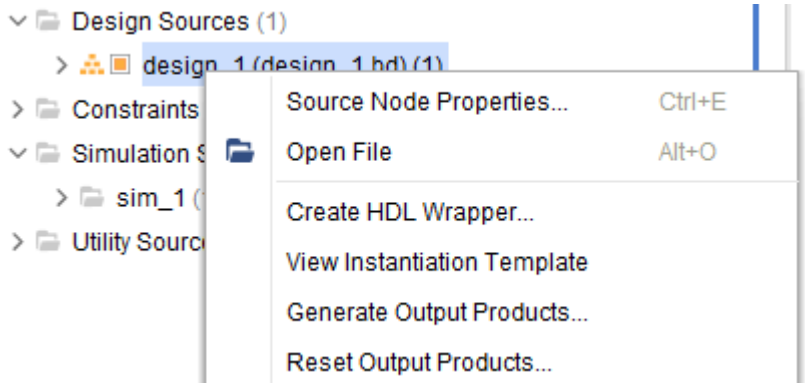


DDR

FIXED_IO

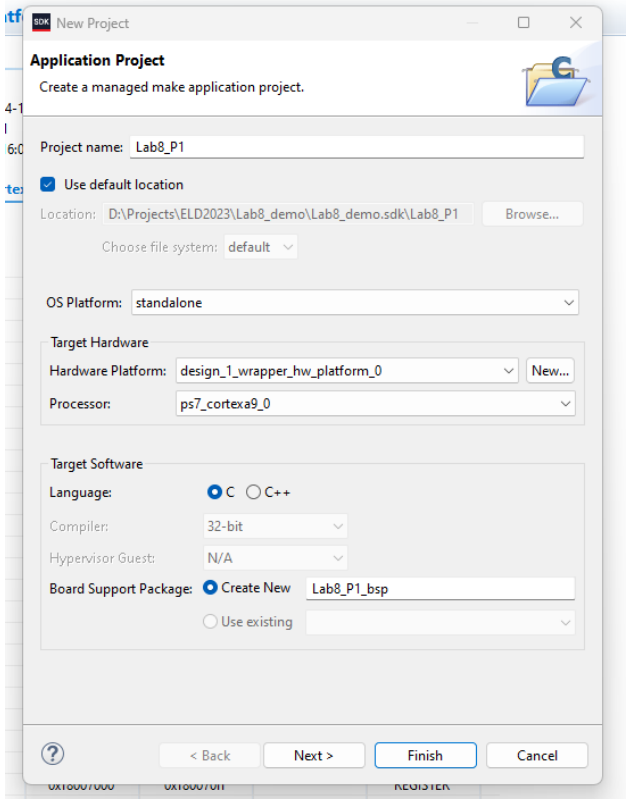
Export the Data and Launch SDK

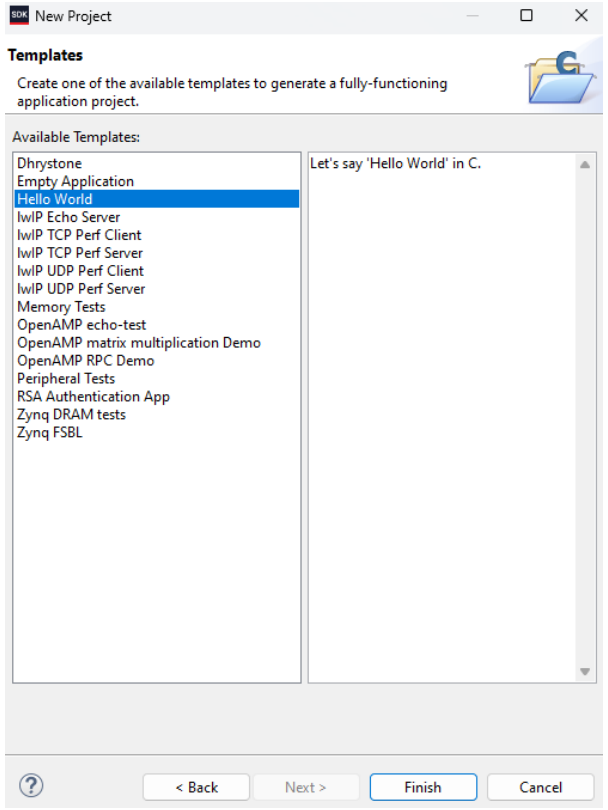
- Create HDL Wrapper and Generate Output Products
- File -> Export Hardware
- File -> Launch SDK



SDK

- Create New Application Project and select Hello World





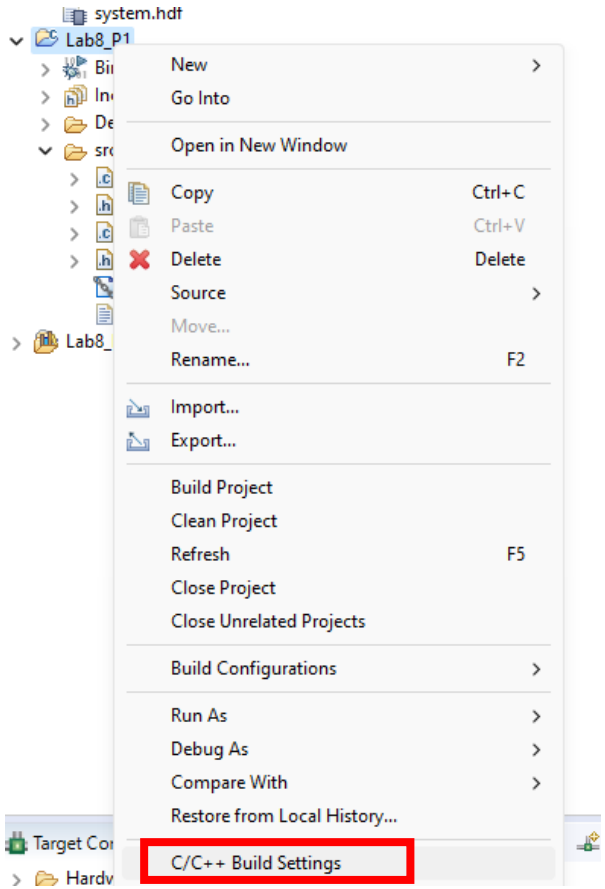
Application Code

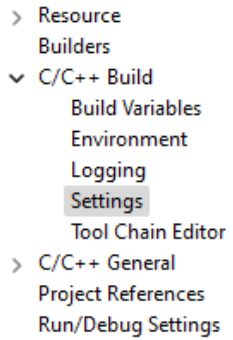
- Update the code

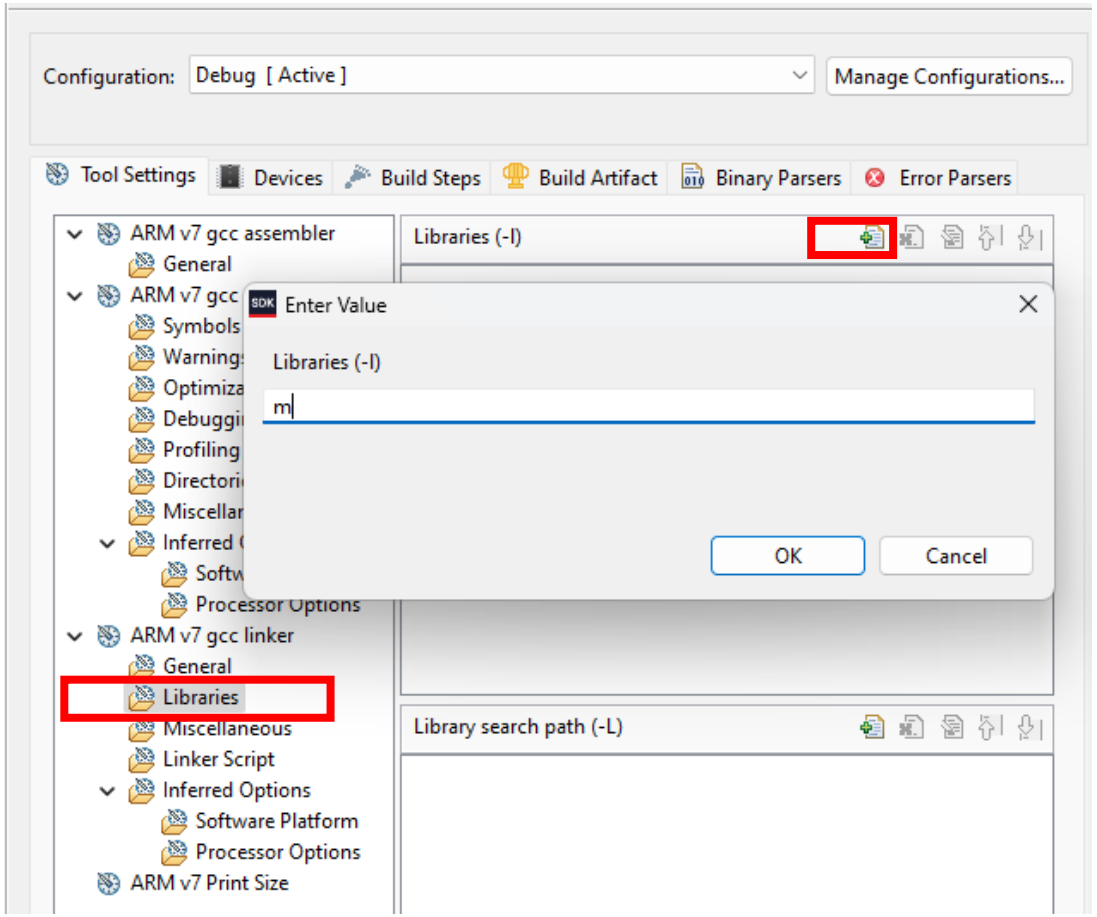
```c
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"


int main()
{
    init_platform();

    print("Hello World\n\r");

    int x=4;
    int n=200;
    int t=2;
    float res1=x/t;
    float res2=(2*log(n))/t;
    float res3=sqrt(res2);
    float result=res1+res3;

    printf("The expression x/t + sqrt(2*ln(n)/t is being calculated\n");
    printf("x/t resulted in %f\n",res1);
    printf("2*ln(n)/t resulted in %f\n",res2);
    printf("sqrt(2*ln(n)/t resulted in %f\n",res3);
    printf("The final result of x/t + sqrt(2*ln(n)/t) in %f\n",result);


    cleanup_platform();
    return 0;
}
```

Application Code

- Add Math library in SDK

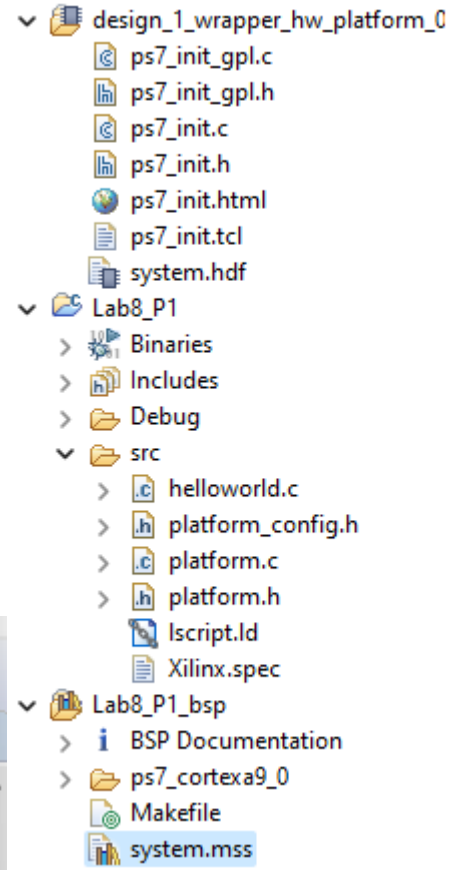


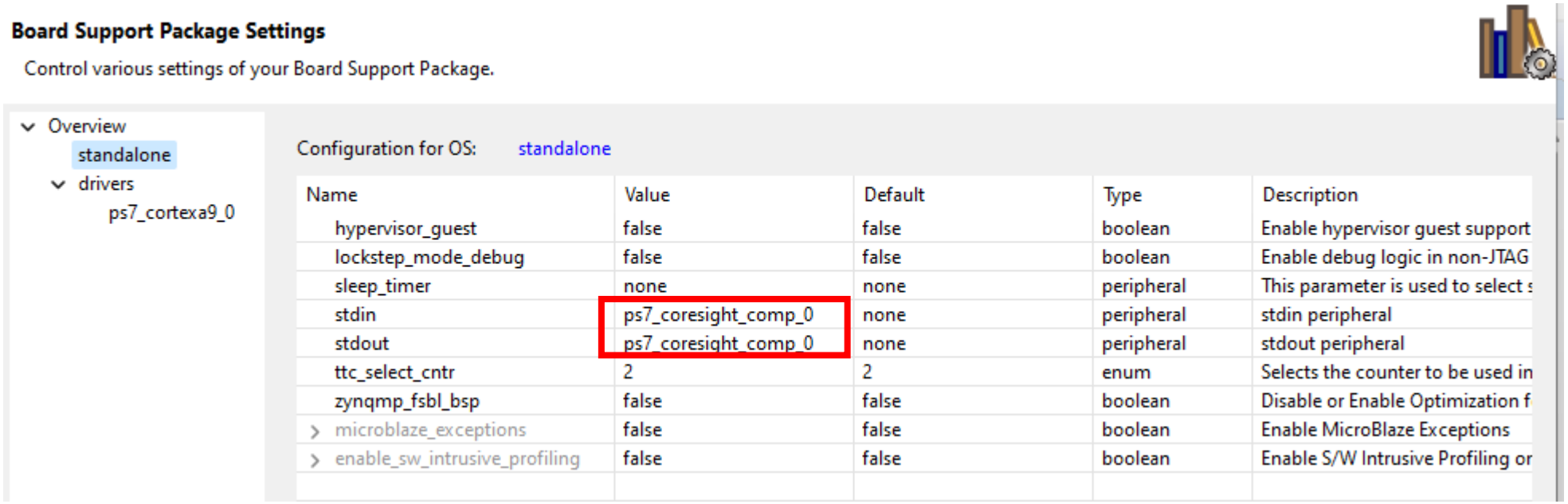




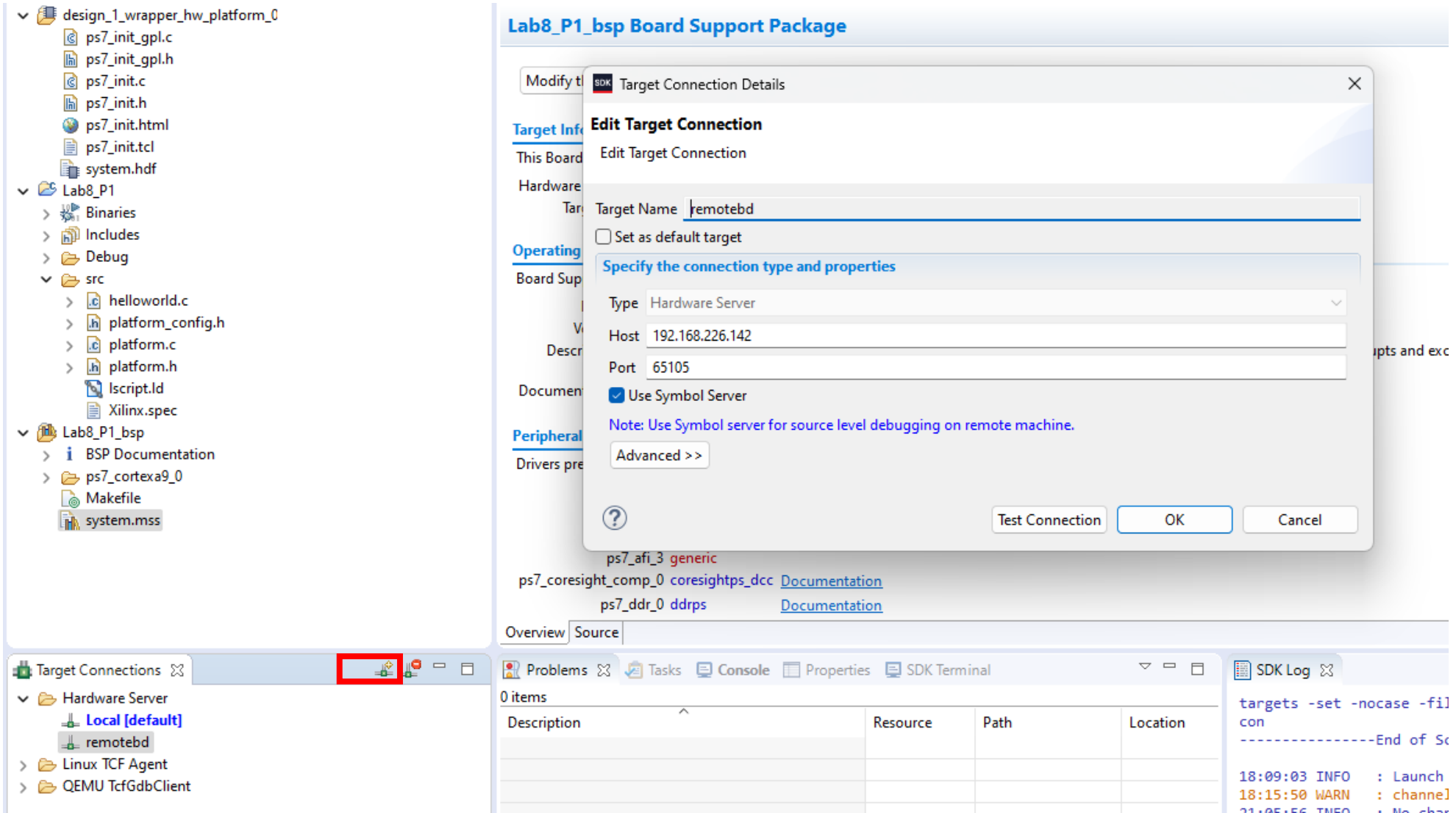
Application Code

- Update the system.mss stdin and stdout

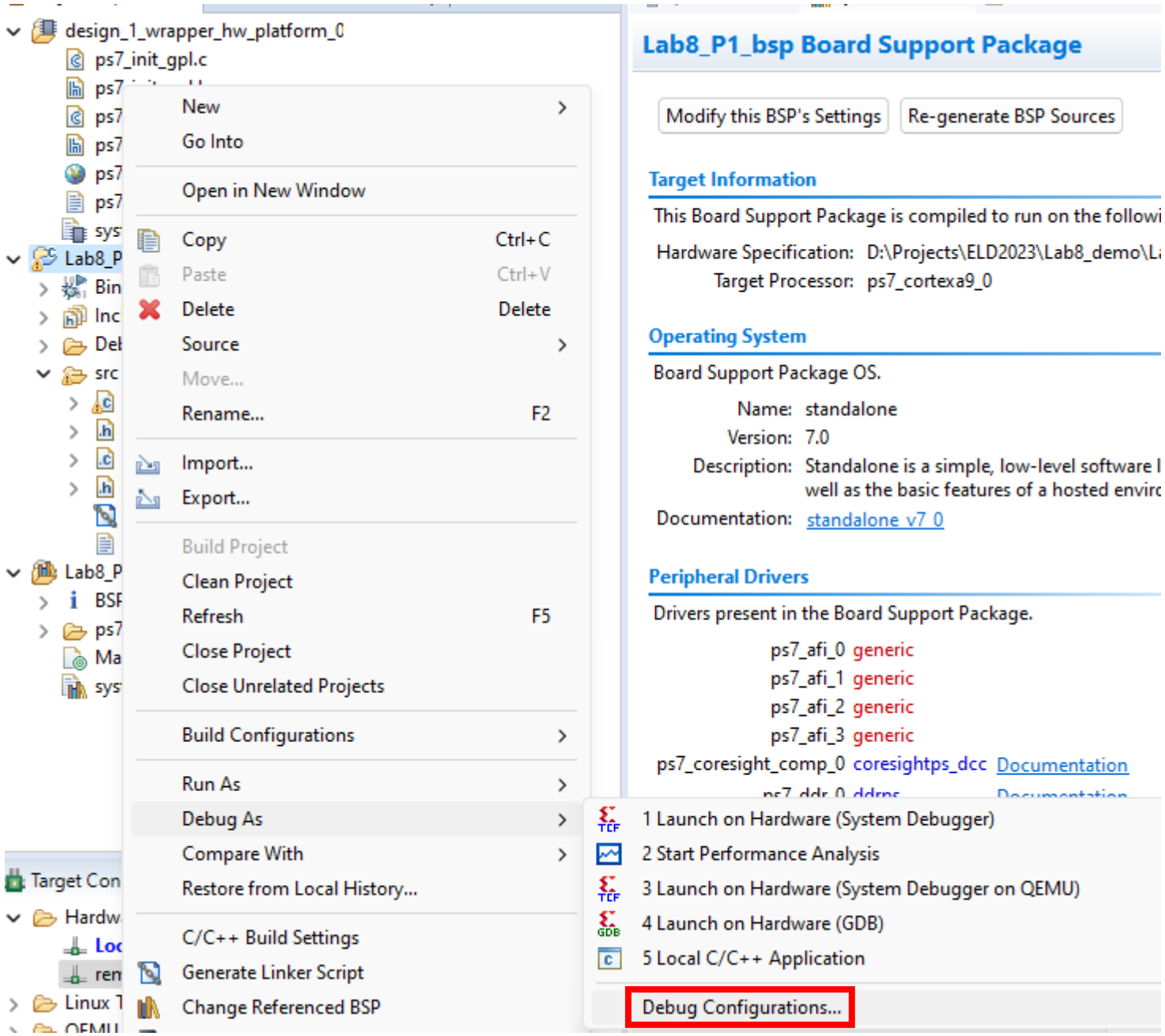




Remote Access

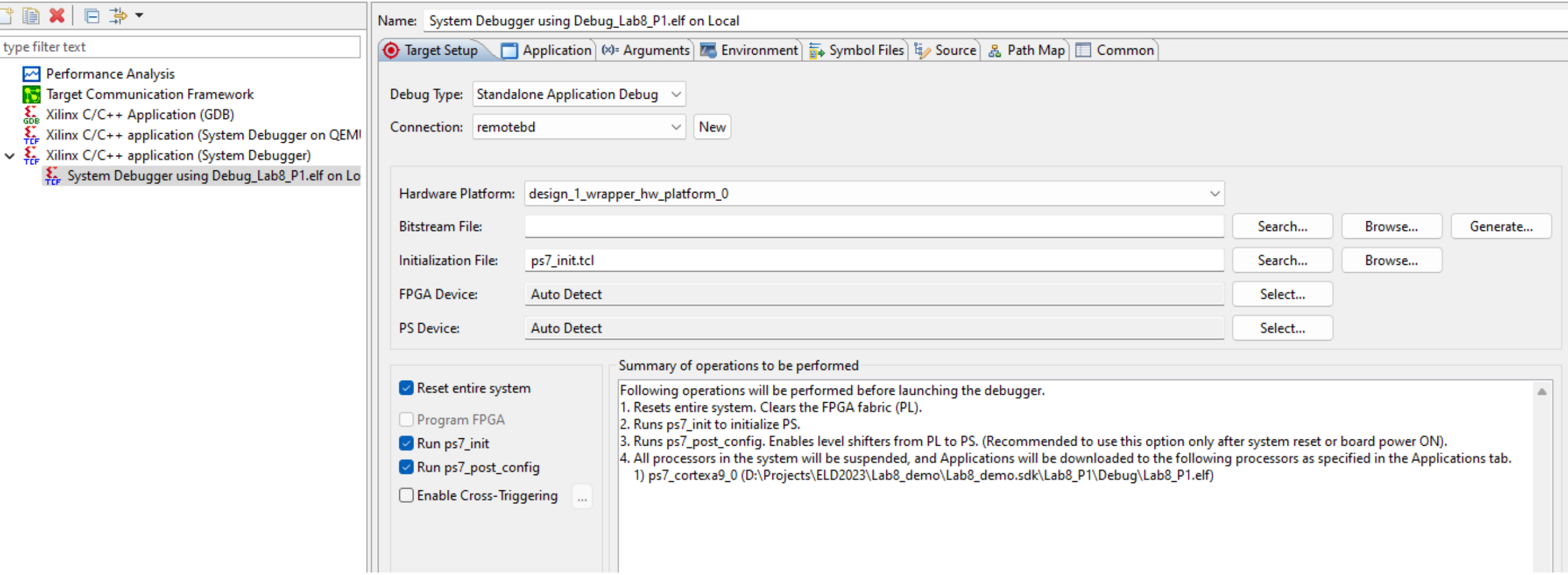


Running Application on Board



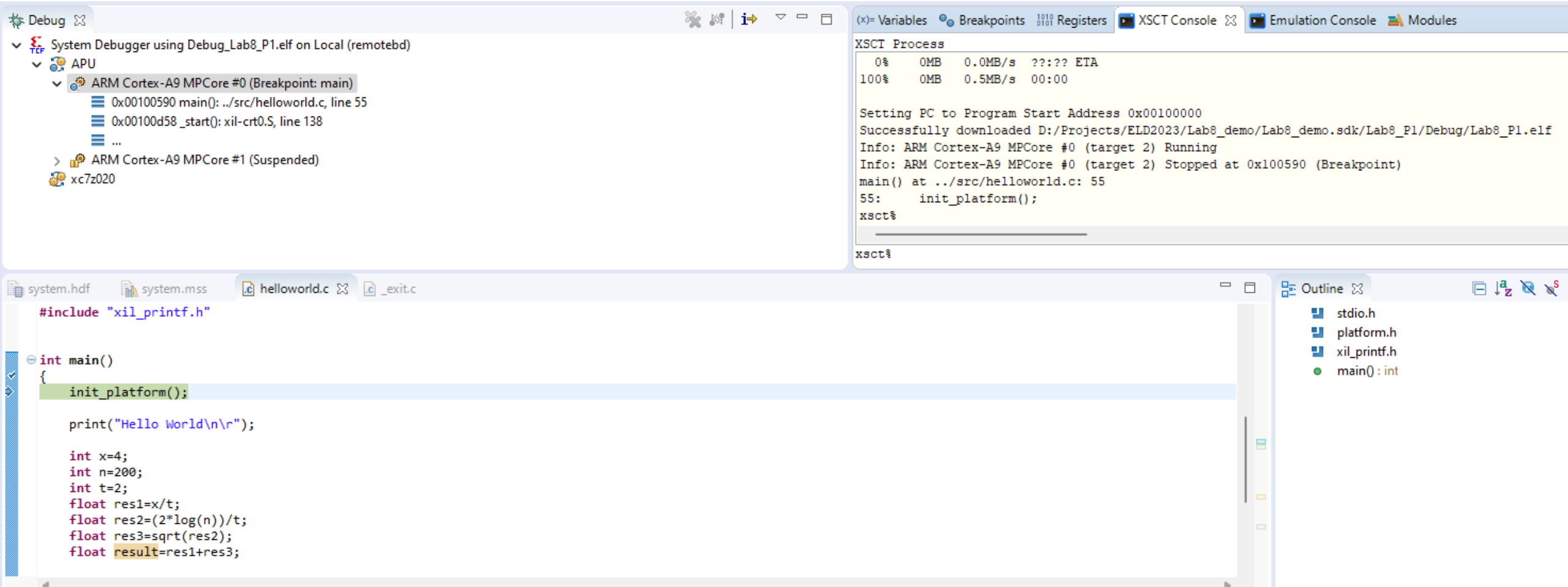
Running Application on Board

- Click on Debug



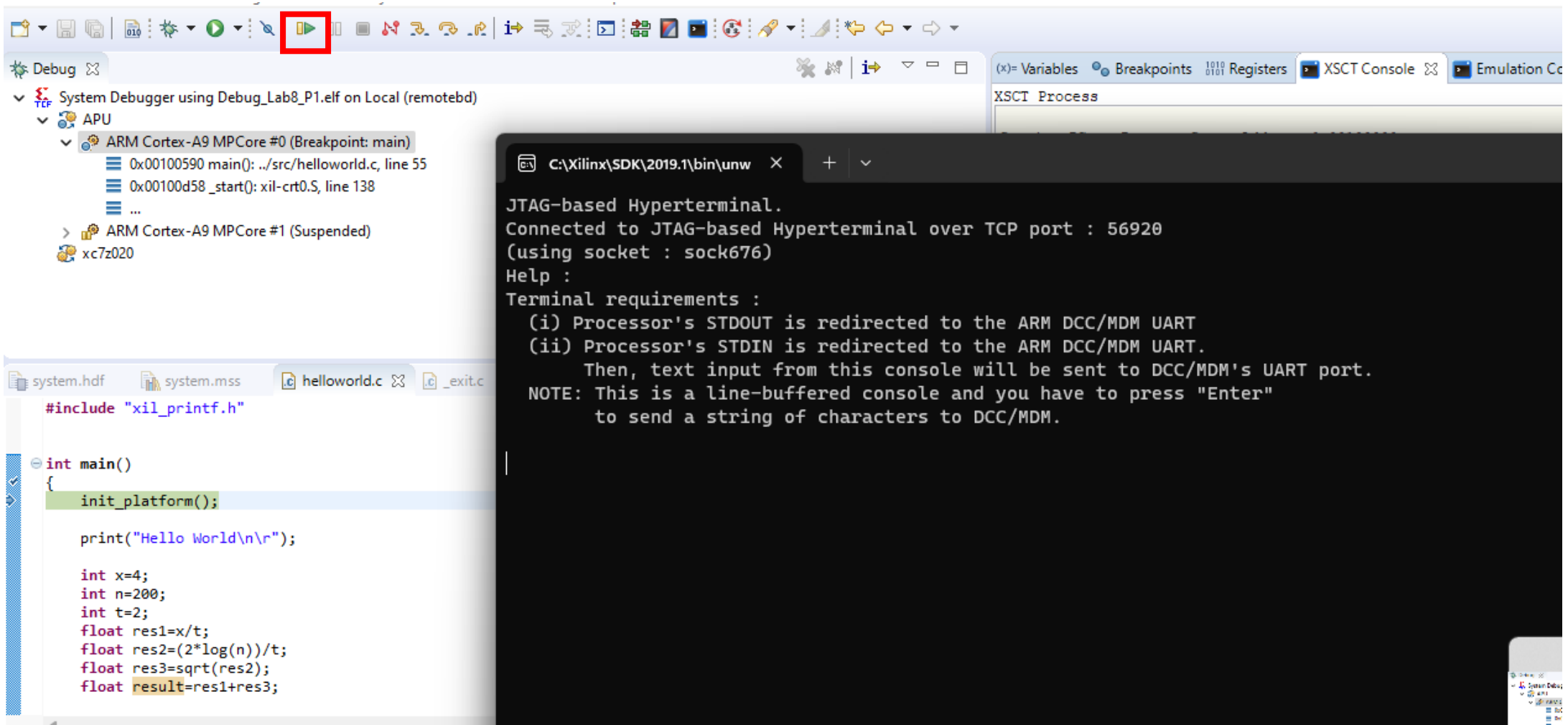
Running Application on Board

- In XSCT Console, type jtagterminal to open the terminal

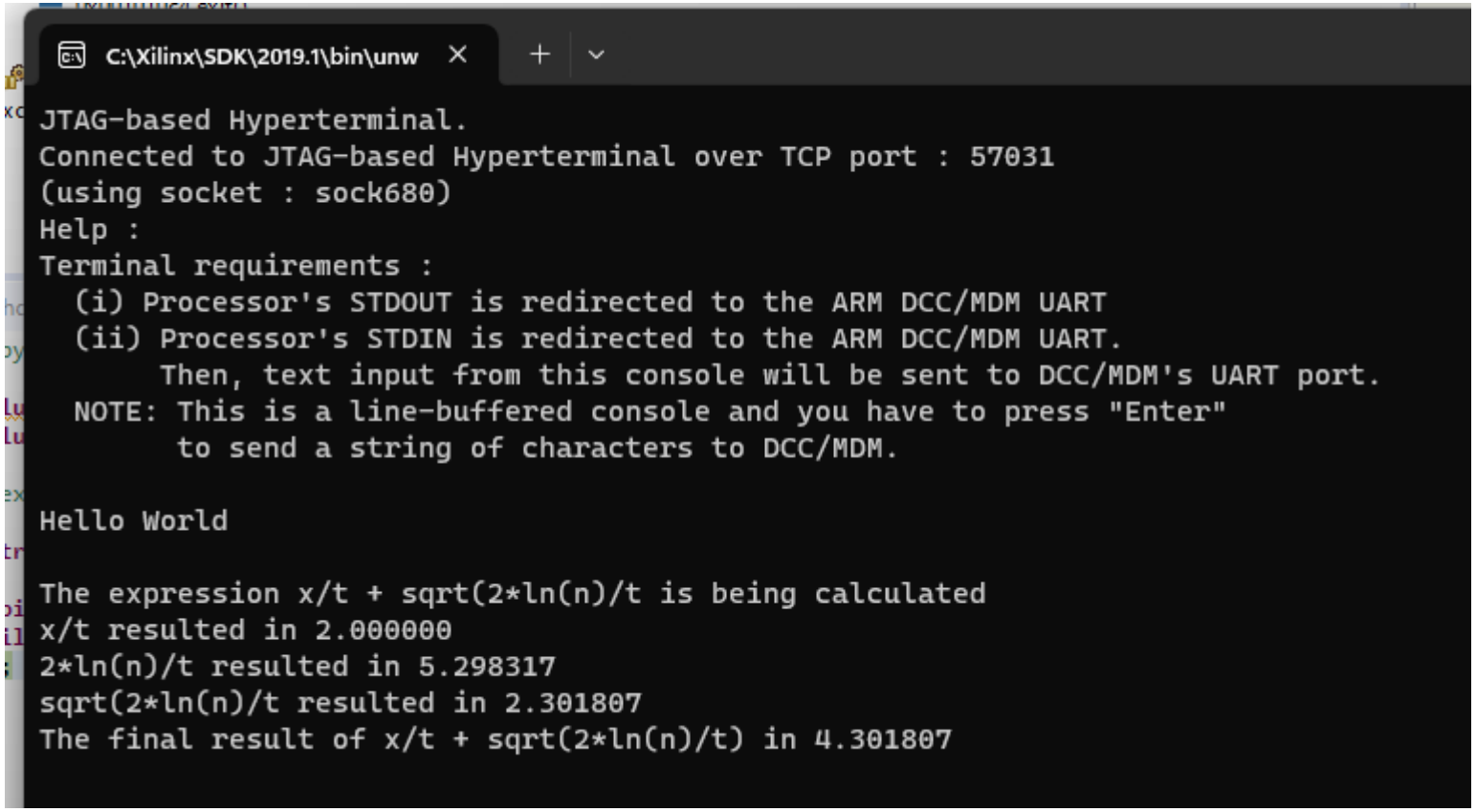


Running Application on Board

- Run the application



Output


C:\Xilinx\SDK\2019.1\bin\unw
JTAG-based Hyperterminal.
Connected to JTAG-based Hyperterminal over TCP port : 57031
(using socket : sock680)
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.
Hello World
The expression x/t + sqrt(2*ln(n)/t is being calculated
x/t resulted in 2.000000
2*ln(n)/t resulted in 5.298317
sqrt(2*ln(n)/t resulted in 2.301807
The final result of x/t + sqrt(2*ln(n)/t) in 4.301807