CO503 : Practical 3 - Multi-Processor System-on-Chip (MPSoC) Design

In this practical, you will use FPGA design tools to create your first Multi-Processor System-on-Chip. A commonly used method to share data between two processors is via shared memory. In Part 1, you will design a simple producer-consumer MPSoC where the inter-processor communication is achieved via shared memory. Subsequently in Part 2 of the practical, you will change the communication method to a dedicated hardware FIFO queue which can achieve better performance compared to shared memory.

Part 1: Producer-Consumer Applications on a Shared Memory Multi-Processor

You will be designing an MPSoC in Qsys with two processor cores. Create a new Qsys system with two Nios II processors, *cpu0* and *cpu1*.

- Each CPU must use its own TIMER and JTAG UART
- Each CPU must use its own ON-CHIP MEMORY device as the instruction memory
- Both CPUs must share a separate ON-CHIP MEMORY device as the data memory

You can use a 50MHz clock signal to drive all components within the Qsys system. Discuss with your classmates how the available data memory can be partitioned between the two CPUs. There should be memory partitions private for each CPU, and one partition shared between the two CPUs for communication. Decide on a partitioning scheme before proceeding.

Once the hardware is complete, create two software projects, one for each CPU. One project will run the producer application, while the other runs the consumer application (producer will produce tokens and send to the consumer, who will consume incoming tokens). You will need to modify the "Linker Script" in the BSP editor for both producer and consumer applications, in order implement the previously decided memory partitioning scheme.

Study the provided code for sample producer and consumer applications. Now you must design a software mechanism to pass data from the producer to the consumer via shared memory. For this, you are given a skeleton code which implements a software FIFO queue, carefully study it first and complete it to place the software FIFO queue within the shared data memory partition. Alternatively, you may write your own software FIFO queue, however it must be implemented within the shared data memory partition.

Get your work checked when complete.

Part 2: Hardware FIFOs

Create a separate MPSoC similar to the one in Part 1, but with the software FIFO mechanism replaced with a dedicated hardware on-chip FIFO memory device in Qsys. Producer and consumer applications running on the two processors should use the new hardware FIFO for communication, instead of the previous software method. Refer to the provided datasheet about using the on-chip FIFO memory component.

Get your work checked when complete.