Welcome to the Webinar Offered by:

National Microelectronics Security Training (MEST) Center

Sponsors:

Introduction to Systems-on-Chip

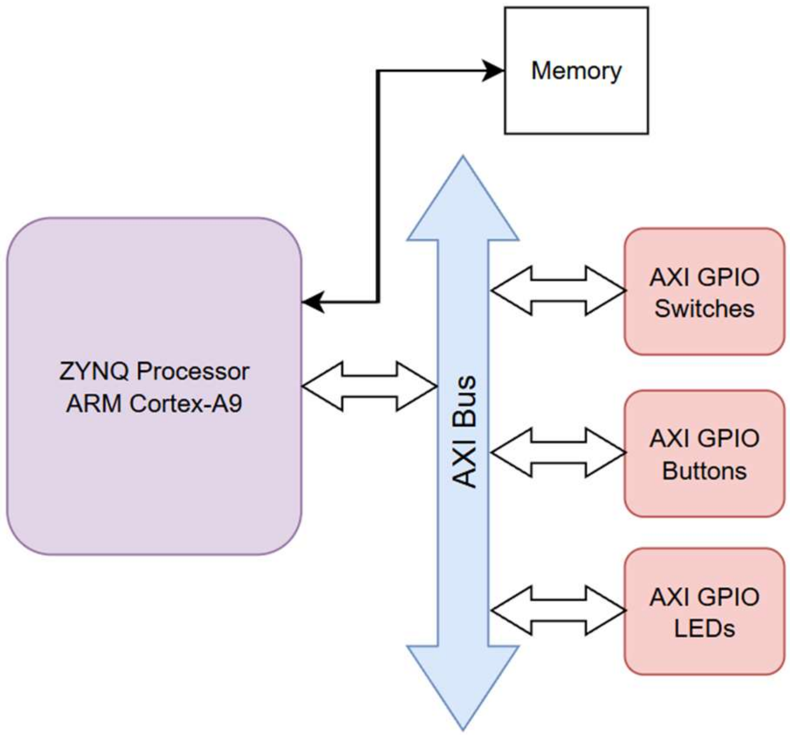
Multi-Processor Integration

Abigail Butka, Kawser Ahmed Muhammed, Wade Fortney,
Christophe Bobda

Agenda

- System-on-Chip Refresher
- Multi-Processor Systems
  - Why Is It Important for Processors to Communicate?
  - How Do Multiple Processors Communicate With Each Other?
  - How Do Processors Communicate Using a Mailbox?
- Case Study and Tutorial
- Conclusion

SoC- Architecture



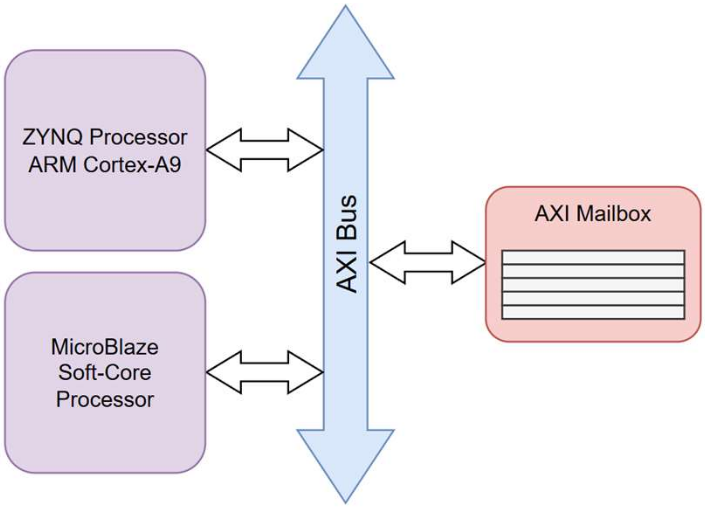
Multi-Processor Systems

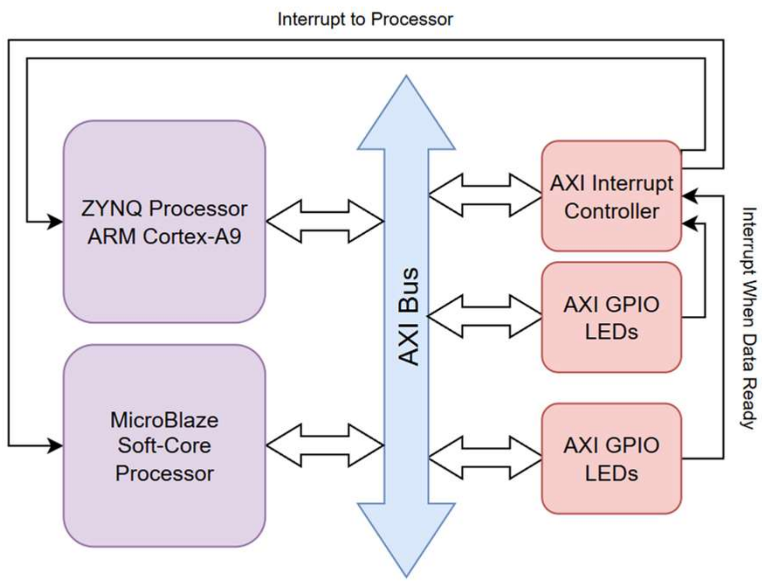
Why Is It Important for Processors to Communicate?

- Parallel processing
  - Allows processors to perform multiple tasks simultaneously to boost performance
- Processor specializations
  - Each processor can be specialized for a specific function, such as high-speed calculations, or high-performance calculations
- Power efficiency
  - Distributing tasks allows for processors to operate at lower power and frequency

How Do Multiple Processors Communicate With Each Other?

- Two main options in an SoC
  1. Indirect communication. Processors read and write data to IPs in the system.
     1. With interrupts
        - When data is ready, the processor A, or an IP, interrupts processor B to let it know to read the data
     2. Without interrupts
        - Processor A writes data to an IP, and Processor B waits for the IP's ready signal
  2. **Direct communication through a mailbox**

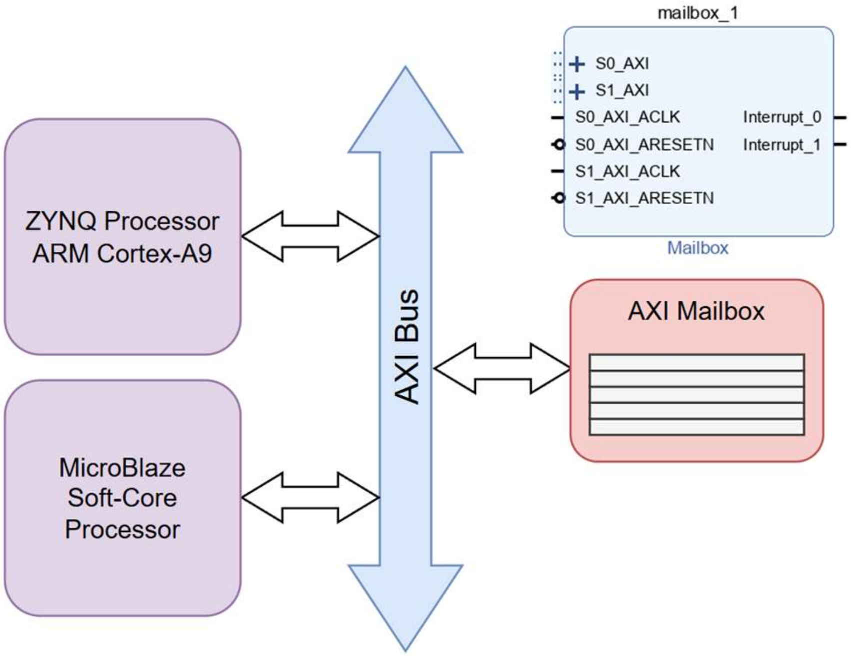




How Do Processors Communicate Using a Mailbox?

- AXI Mailbox
  - Allows communication between only two processors
  - Stores data in memory using a FIFO (First-In, First-Out)

- For the MicroBlaze to send data to the ZYNQ processor…
  - The MicroBlaze will write data to the Mailbox
    - XMbox_WriteBlocking(Mailbox_Address, data, num_bytes);
      - This command is blocked if the mailbox is full
      - The writing processor will wait for the mailbox to have space before continuing
  - The Mailbox receives this data and stores it in the FIFO
  - Meanwhile, the ZYNQ is waiting for data to enter the Mailbox
    - XMbox_ReadBlocking(Mailbox_Address, data, num_bytes);
      - Similarly, the reading processor will wait for the mailbox to have a message before continuing
  - When the MicroBlaze is done writing to the Mailbox, the Mailbox can respond to the ZYNQ's data read request.



What Processors Can We Use in Systems-on-Chip?

- Hard processors
  - Many Systems-on-Chip and FPGAs come with “hard” processors built into the fabric

  - Xilinx Zynq - ARM Cortex-A9
  - Xilinx Zynq Ultrascale+ - ARM Cortex A53 and Cortex-R5
  - Intel Stratix 10 – ARM Cortex-A53

- Soft processors
  - When SoCs and FPGAs do not have a hard processor, or require more processors, “soft-core” processors can be used.
  - These are microprocessors made up of code that will be implemented into the fabric

  - Xilinx MicroBlaze
  - Intel Nios II and NIOS V
  - RISC-V

Case Study and Tutorial