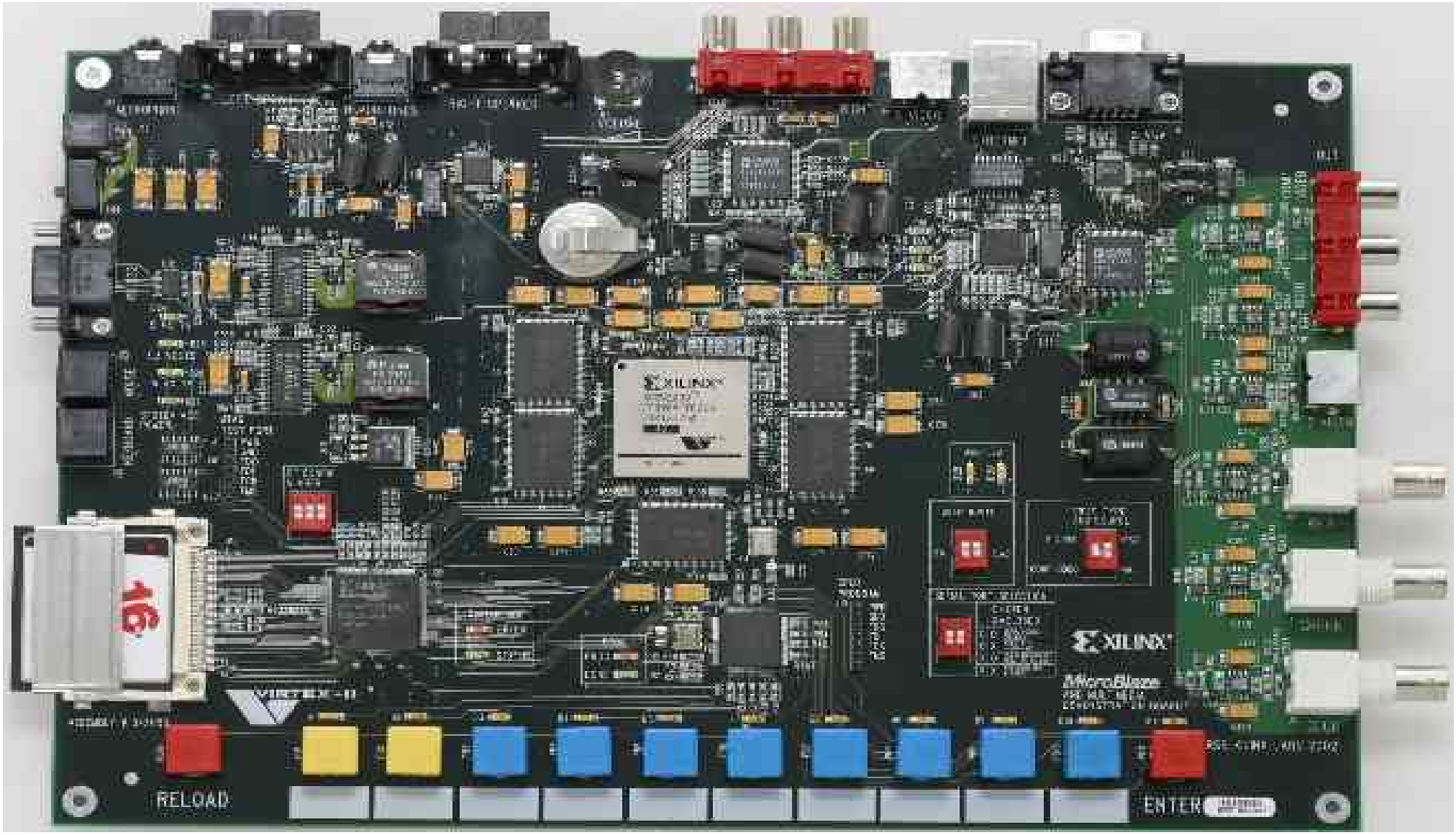
An Introduction to Xilinx System Level Prototyping Station (SLPS)

By: Amir Ali Yazdanshenas
Supervisor: Dr. Khalid



Overview

1- MicroBlaze™ Features & Applications
2- MicroBlaze™ Hardware Structure
3- IP Cores
4- Design Flow & CAD Tools
5- Debug Tools

6- Short Demo!

MicroBlaze™ Features and Applications

- Supports Real-Time PAL/NTSC TV I/O
  - Single channel - Composite/S-Video/ RGB format
  - 10bit ADC/DAC with 13.5/27MHz Pixel Rate
  - True Color SVGA output to monitors

→ Image Processing

- Audio CoDec with Power Amplifier (18 Bit ADC/DAC)
  - One Stereo + Two Mono + Microphone inputs
  - Volume/Mute Control, Speaker Output Drive

→ Audio Processing

- 10/100 Full/Half Duplex Ethernet Adapter

→ Networking

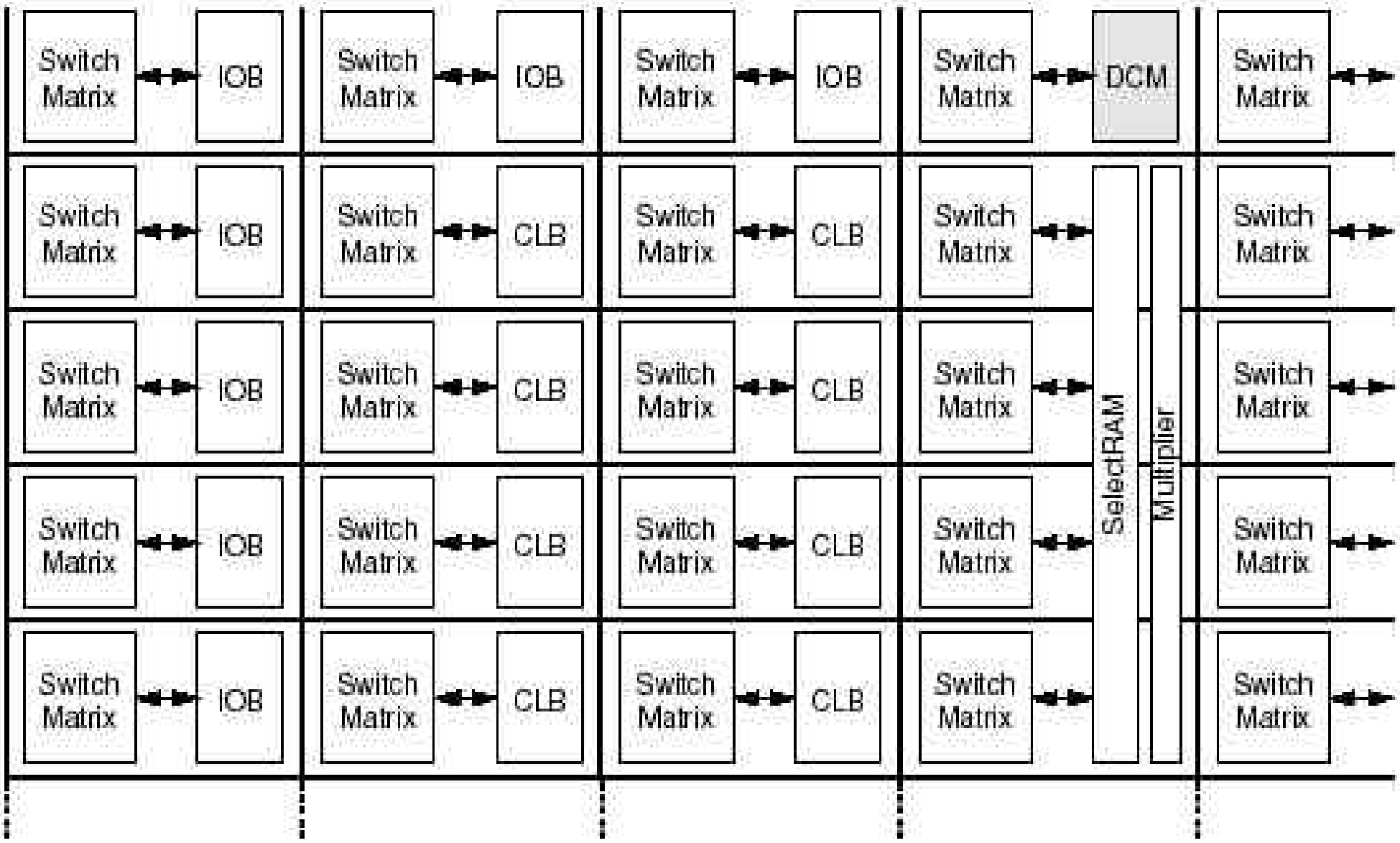
- General I/O Interface
  - RS-232 Serial Interface/User Buttons & Switches
  - Keyboard/Mouse
- (+)
  - 5 Banks of 512k x 32 SRAM
  - Xilinx Virtex-II FPGA

→ Any SOC/ Embedded System!

Xilinx Virtex-II™ Specifications

- Industry Platform SRAM based FPGA
- 2M System gates @ 450MHz+
- 1Mb Dual Port RAM
- SRAM/DRAM Interface
- 56 Dedicated 18-bit x 18-bit multiplier blocks
- 8 Digital Clock Management Circuitry (x1.5 – x16)
- Fourth Generation Segmented Routing Architecture
- SelectI/O™Ultra technology (DCI & SE/DE)

Xilinx Virtex-II Array Overview



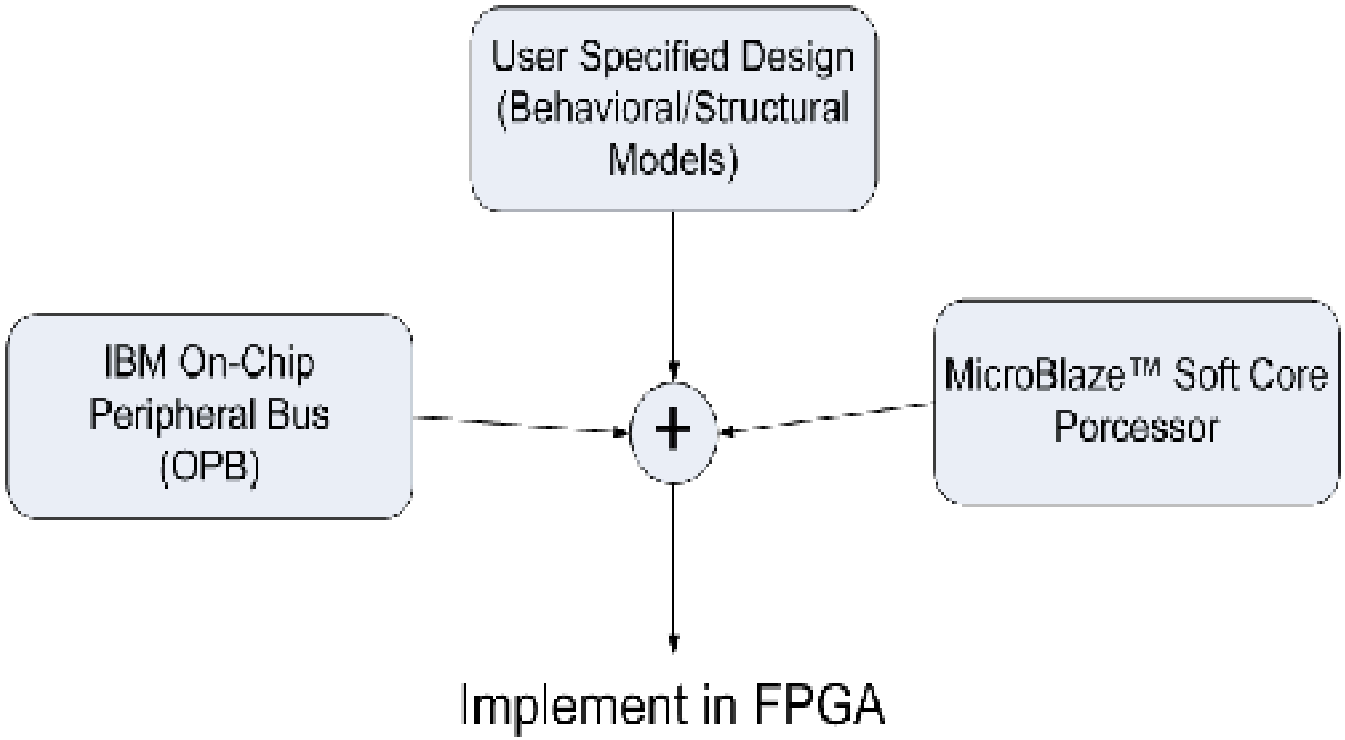
Design Customization

1- Describe your hardware:

- Structural Model
- Behavioral Model

2- Add library components/IP cores to your design to control the Input/Output using:

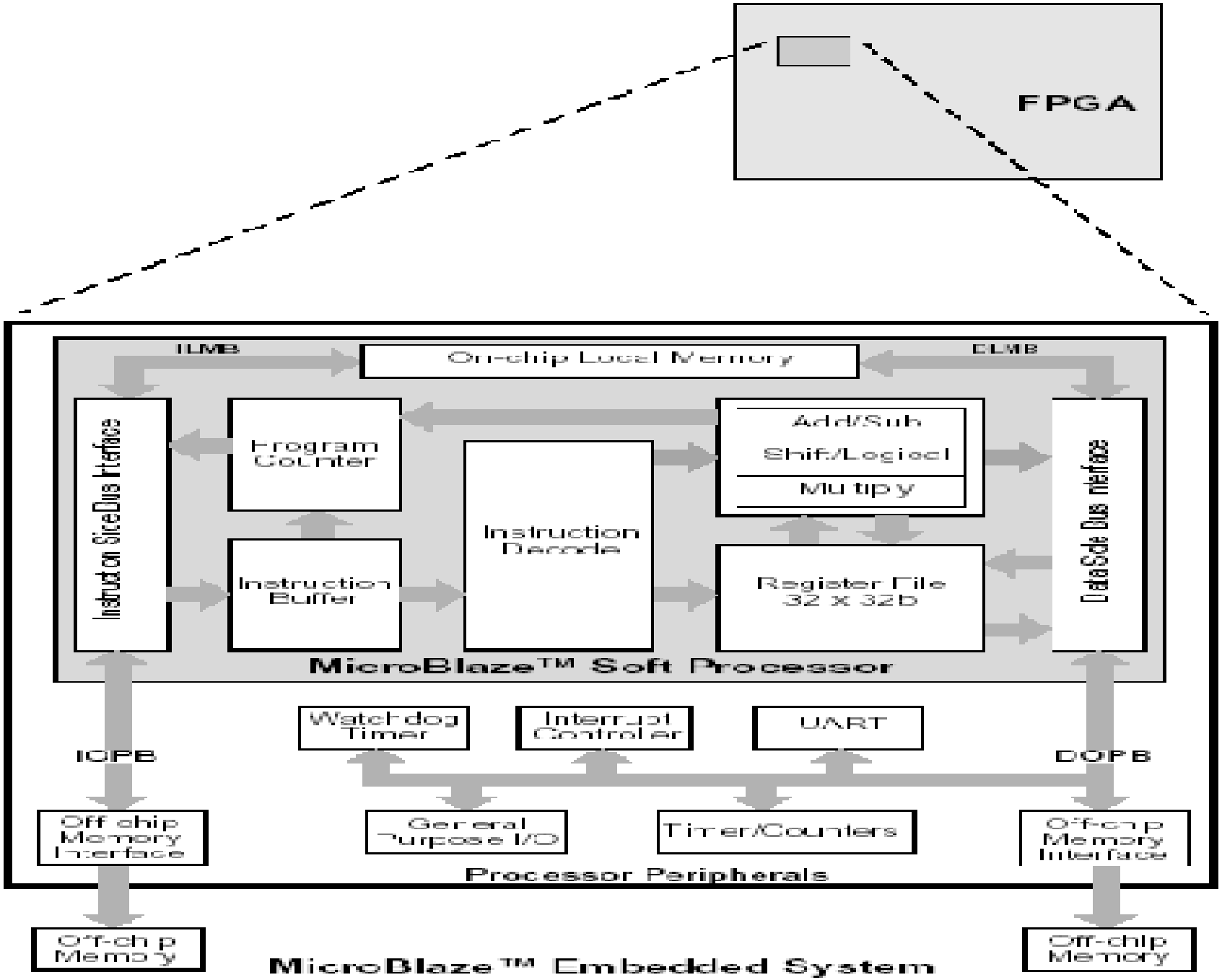
- MicroBlaze™ Soft Core Processor
- IBM On-chip Peripheral Bus (OPB)



MicroBlaze™ Embedded Soft Core

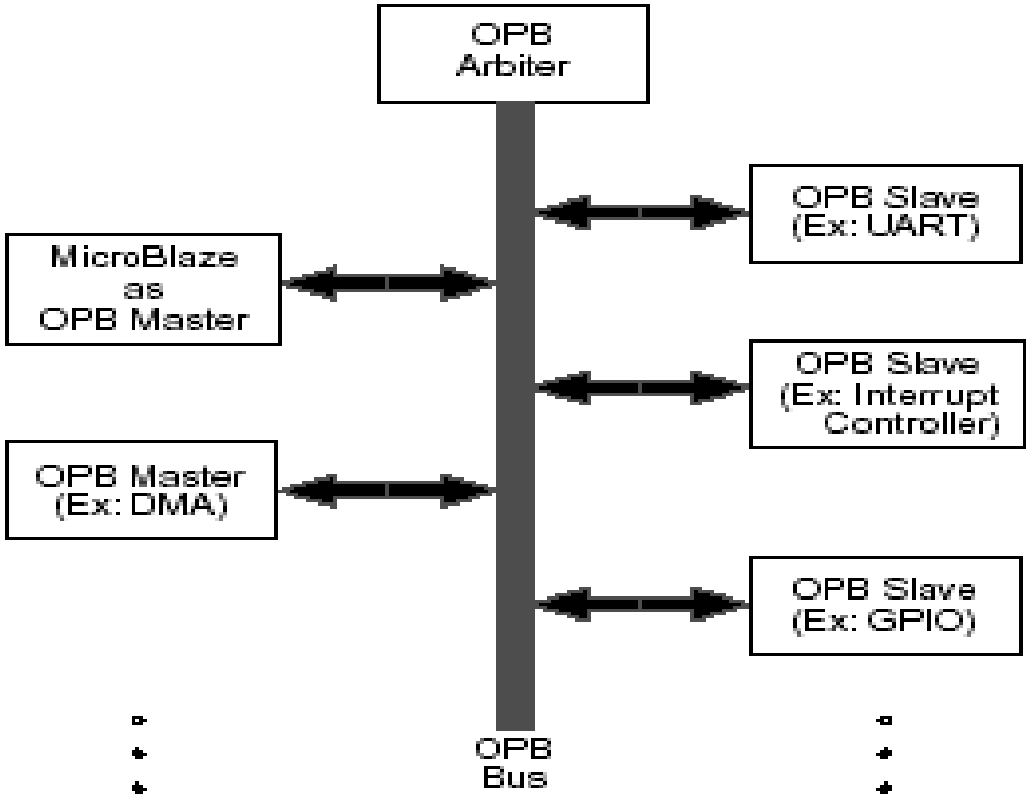
- MicroBalze™ is just a Processor Soft IP Core
- 32-bit pipelined RISC architecture optimized for implementation in FPGAs (@125MHz / 900LC)
- Harvard Architecture
- Supports IBM CoreConnect™ On-chip Peripheral Bus (OPB)
- Supports Local Memory Bus (LMB) for fast access to on-chip RAM + external memory bus connection

MicroBlaze Embedded System



On-Chip Peripheral Bus (OPB) Overview

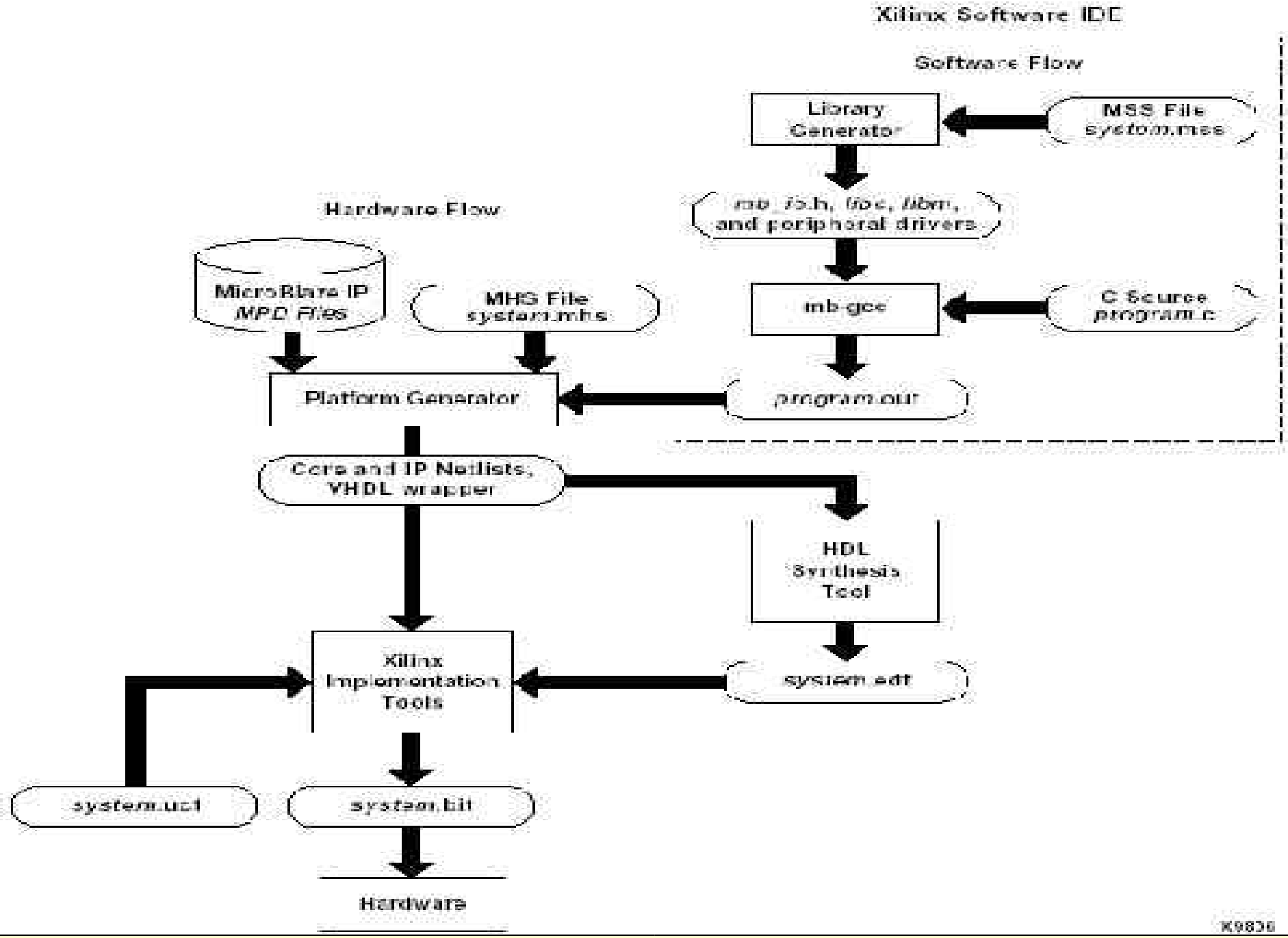
- Standard IP Core Bus for embedded systems introduced by IBM
- Provides easy connection to peripheral devices (e.g. Timer/UART/…)
- Fully synchronous, 32-bit address/data bus
- Single cycle transfer between master and slave
- Byte/Half word/Word transfer
- No tri-state drivers required



Design Development Flow

- **Step 1**
  - Define your hardware using Verilog/VHDL.
  - Include standard OPB signals in your design.
- **Step 2**
  - Instantiate your design along with other peripherals inside **MHS** file.
  - Use **Platform Generator** to compile your design.
- **Step 3**
  - Define your software configuration within the **MSS** file.
  - Use Library Generator to compile.
- **Step 4**
  - Use **Project Navigator** to generate FPGA downloadable bit stream.
- **Step 5**
  - Download your design to FPGA by using **iMPACT** or **SystemACE**.
- **Step 6**
  - Debug your design!

Design Development Tool Flow



Microprocessor Hardware Specification (MHS)

- Each peripheral connected to the OPB is declared in the MHS file.
- Essentially the MicroBlaze processor would be the Master device.
- User should specify the followings inside the *.mhs* file:
  - One of six configurations of the MicroBlaze Bus interface
  - Peripherals (+ your design)
  - Connectivity of the System
  - Address Space

```
SELECT SLAVE opb_uartlite
CSET attribute HW_VER = 1.00.a
CSET attribute INSTANCE = myuartlite1
CSET attribute C_HIGHADDR = 0xFFFF80FF
CSET attribute C_BASEADDR = 0xFFFF8000
CSET signal RX = rx1
CSET signal TX = tx1
CSET signal OPB_Clk = sys_clk
CSET signal Interrupt = int_periph, PRIORITY=1
END
SELECT MASTER microblaze
CSET attribute HW_VER = 1.00.a
CSET attribute INSTANCE = microblaze1
CSET attribute CONFIGURATION = 1
CSET signal Interrupt = Interrupt
CSET signal Clk = sys_clk
CSET attribute C_LM_HIGHADDR = 0x00007fff
CSET attribute C_LM_BASEADDR = 0x00000000
END
```

Microprocessor Software Specification (MSS)

- The *.mss* file (supplied by user) specifies:
  - Standard I/O devices + drivers
  - Interrupt handler routines
  - Boot devices
  - Debug modules
  - …

```
SET attribute BOOT_PERIPHERAL = my_jtag
SET attribute DEBUG_PERIPHERAL = my_jtag
SET attribute XMDSTUB = code/xmdstub.out
SET attribute BOOTSTRAP = code/bootstub.out
SET attribute EXECUTABLE = code/hello_world.out
SET attribute STDIN = my_uartlite
SET attribute STDOUT = my_uartlite
SELECT INSTANCE my_uartlite
CSET attribute DRIVER_VER = 1.00.a
CSET attribute DRIVER = uartlite
CSET attribute LIBRARY = XilFile
CSET attribute INT_HANDLER = my_uartlite_hndl, Interrupt
END
```

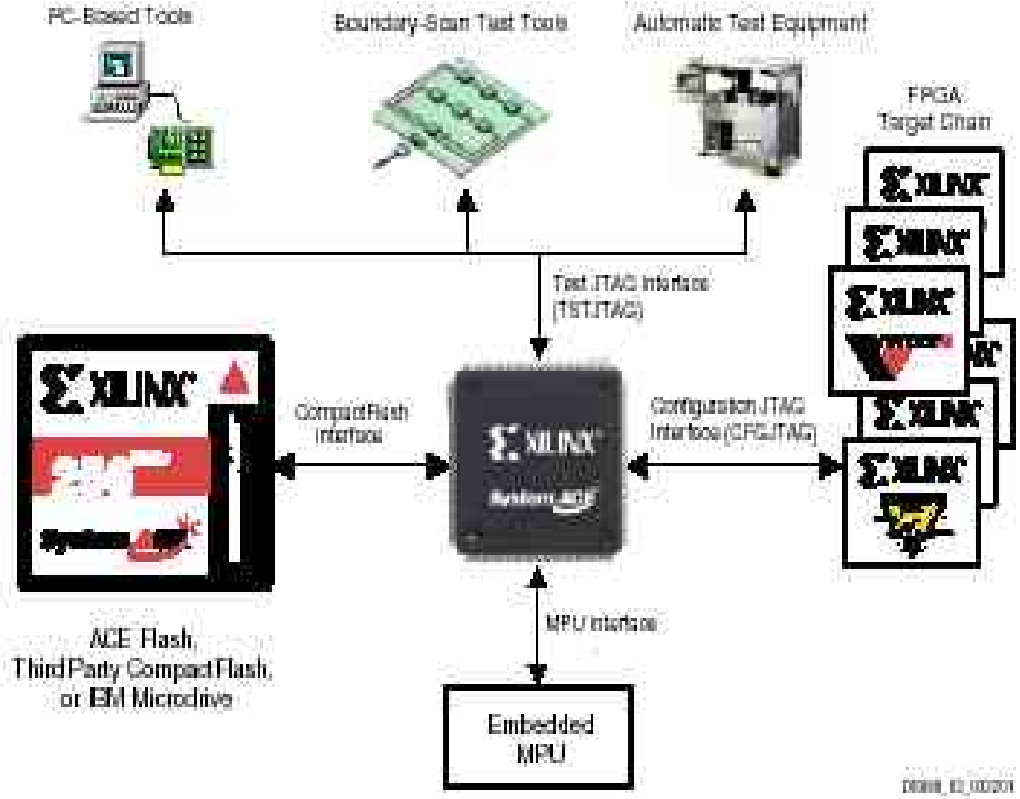
Implementation Tools

Implementation Includes

- Net list generation
- Placement
- Routing

→ Xilinx ISE or XPS

- Generating configuration bit-stream
  - Downloading bit-stream → Xilinx ISE (iMPACT)
    - JTAG Cable Connection
    - ACE Compact Flash Card



Debug Tools - Software Debugging

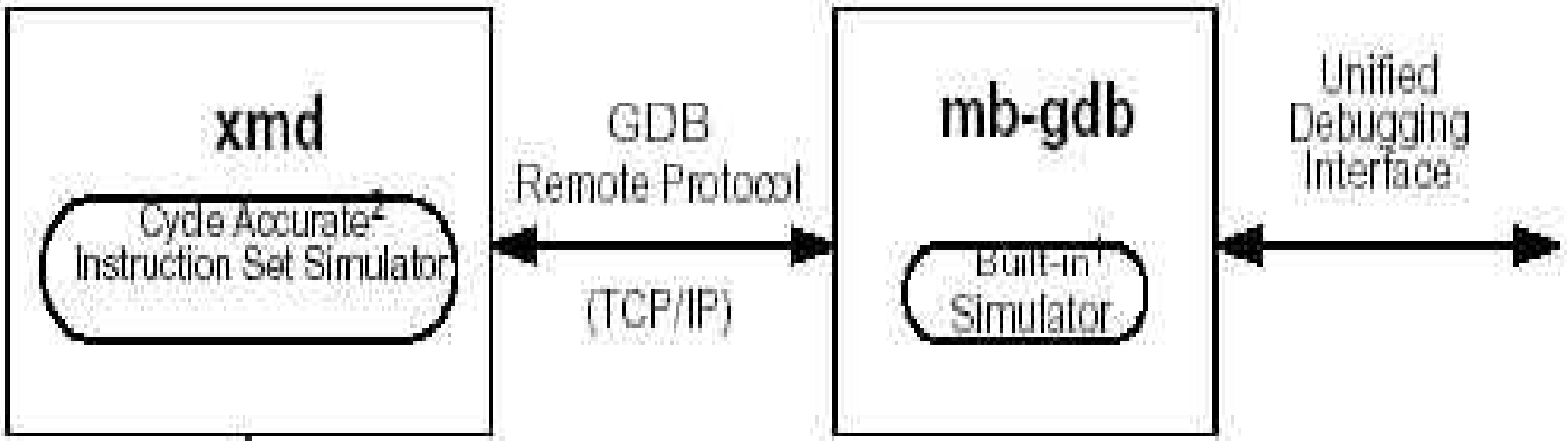
1- Using Simulator

- Instruction Set Simulator (ISS)

  Non-intrusive, uses a model of instruction set to execute the application program. It's been integrated into GNU's GDB.
  (e.g.) *$> mb-gcc –g → mb-gdb*

- Cycle Accurate ISS

  Non-intrusive, uses detailed information of processor's pipeline behavior to model instruction execution (clock cycle resolution). XMD is the engine but the interface is still *mb-gdb*.
  (e.g.) *$> mb-gcc → xmd -t sim*

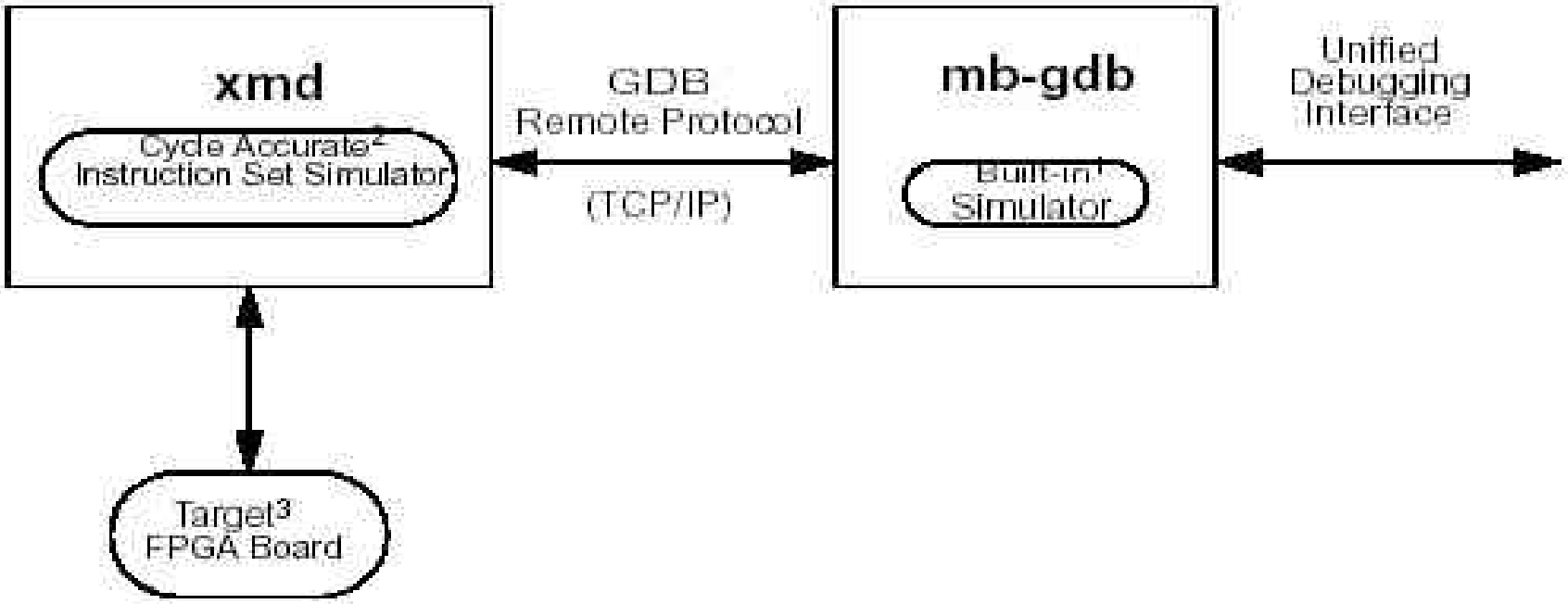


Debug Tools - Software Debugging (cont'd)

2- Using hardware board

Intrusive, a debug agent *(xmdstub)* is compiled along with the code which controls the execution of the program, accessing registers/memories, and provides communication between the host and the board.

(e.g.)*$> mb-gcc -g -x1-mode-xmdstub prog.c → xmd -t hw → mb-gdb*



Debug Tools – Hardware Debugging

- Hardware Simulation
  Platform Generator generates the HDL models of the processor and the peripherals. These models are simulated using ModelSim.

- Co-simulation
  Debugging hardware and software concurrently using simulation. Platform Generator generates both the HDL models and configuration models for local memories. ModelSim simulates the interaction between the hardware and the program by tracing important signal waveforms, registers and memory contents.

Conclusions:

- Xilinx System Development Kit provides the best set of EDA/CAD tools to get hands-on experience on rapid system prototyping based FPGAs, Hardware Emulation/Software Simulation environments.
- Provides flexible and versatile hardware substrate to develop Embedded Systems or IP Cores (hardware/software).

References

- "Embedded System Tools Guide", Xilinx Inc., EDK 6.1, 2003.
  http://www.xilinx.com/ise/embedded/est_guide.pdf
- "MicroBlaze Hardware Reference Guide", Xilinx Inc., 2002.
  http://www.xilinx.com/ipcenter/processor_central/microblaze/doc/hwref.pdf
- "MicroBlaze Software Reference Guide", Xilinx Inc, 2002.
  http://www.memec.co.jp/html/xilinx/ eboard/docs/xilinxmicroblaze/swref.pdf
- "Virtex-II Platform FPGA User Guide", Xilinx Inc., 2002.
  http://www.xilinx.com/bvdocs/userguides/ug002.pdf
- "ISE 6 In-Depth Tutorial", Xilinx Inc., 2003.
  http://direct.xilinx.com/direct/ise6_tutorials/ise6tut.pdf