

EE3220 System on Chip Design

Lecture Note 10

FPGA based SoC

Review & Preview

- Lecture 1: Course Overview
- Lecture 2: Intro to Modern Microprocessor (CPU Architecture)
- Lecture 3: Embedded System Software (Software Architecture)
- Lecture 4: ARM Cortex-M4 (ARM Hardware)
- Lecture 5: From C to ARM Assembly (C Language, ARM Firmware)
- Lecture 6: Team Based Learning (USB Firmware)
- Lecture 7: AAPCS (ARM Firmware)
- Lecture 8: Exceptions & Interrupts (ARM Firmware)
- Lecture 9: Intro to HDL & FPGA Design (HDL Design)
- **Lecture 10: FPGA based SoC (FPGA based SoC)**
- Lecture 11: SoC Integration & Emerging ISA (AMBA, RISC-V)
- Lecture 12: Team Based Learning (Wi-Fi SoC)

Hong Kong RISC-V Development Summit

CityUHK + StarFive Joint Lab Establishment



The event was attended by Hong Kong Financial Secretary (Paul Chan) and Hong Kong Investment Corporation (HKIC) Chair (Clara Chan). Professor Ray was one of the keynote speakers and forum panelists.



Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

FPGA Use Cases

- (1) FPGA based Embedded Systems
- (2) FPGA Acceleration Cards
- (3) ASIC Prototyping with FPGA
- (4) FPGA Emulation Platform

(1) FPGA based Embedded Systems

- Telecommunication
 - Ex: Software-Defined Radio
- Computing
 - Ex: Edge Computing
- Automotive
 - Ex: Advanced Driver Assistance Systems – ADAS
- Multimedia
 - Ex: Real-Time Video Processing
- IoT Applications
 - Ex: Robotic Control System



(1) FPGA based Embedded Systems: AMD Xilinx Boards

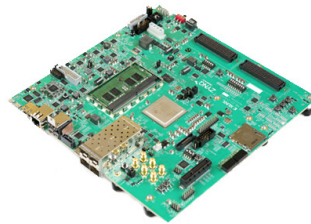
**Zynq-7000 SoC
ZC702**



**Zynq-7000 SoC
ZC706**



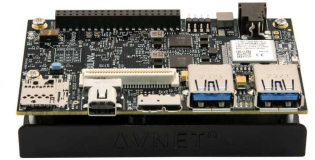
**Zynq UltraScale+ MPSoC
ZCU102**



**Zynq UltraScale+ MPSoC
ZCU104**



Avnet Ultra96-V2



(2) FPGA Acceleration Cards: PCIe based SmartNIC

- High Performance Computing (HPC)
 - Scientific Calculation (Weather Forecast)
 - Financial Calculation (Risk Assessment)
- Network Acceleration
 - TCP/IP Offloading (High Speed Trading)
 - Data Encryption (Cybersecurity)



(2) FPGA Acceleration Cards: AMD Xilinx Alveo Family

Alveo U200



VIRTEX UltraScale+

1,182k LUTs

Dual slot, full height

64GB DDR, 77GB/sec

PCIe Gen3

2x QSFP 28 (100GbE)

< 225W

Alveo U250



VIRTEX UltraScale+

1,728k LUTs

Dual slot, full height

64GB DDR, 77GB/sec

PCIe Gen3

2x QSFP 28 (100GbE)

< 225W

Alveo U280



VIRTEX UltraScale+

1,304k LUTs

Dual slot, full height

8GB HBM2, 460GB/sec

PCIe Gen3, Gen4, CCIX

2x QSFP 28 (100GbE)

< 225W

Alveo U50



KINTEX UltraScale+

872k LUTs

Single slot, half height

8GB HBM2, 316GB/sec

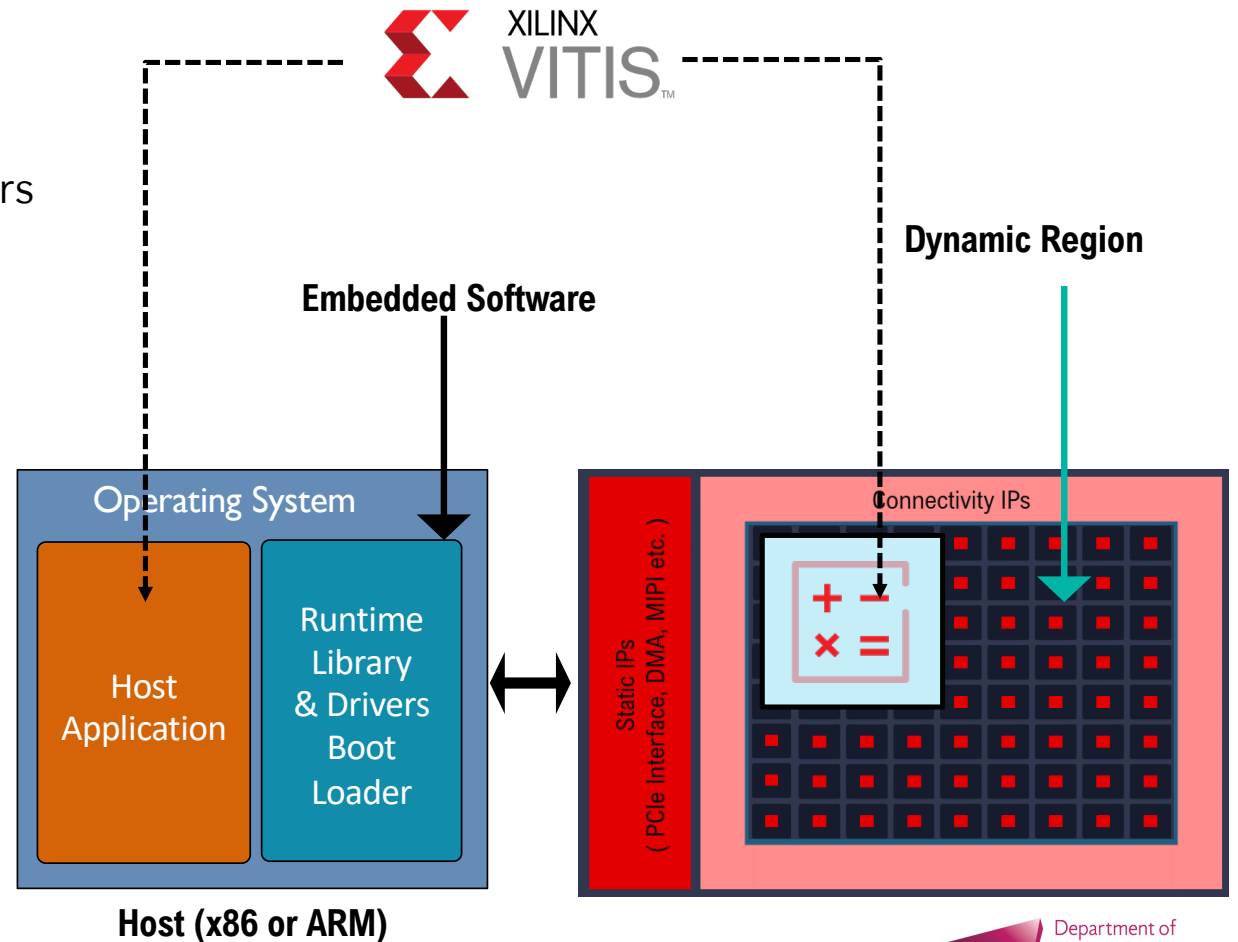
PCIe Gen3, Gen4, CCIX

1x QSFP 28 (100GbE)

< 75W

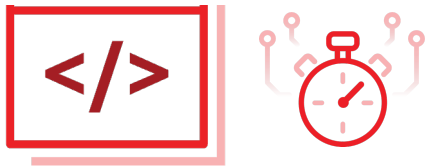
(2) FPGA Acceleration Cards: Servers / Workstations

- PCIe Accelerator Cards:
 - PCIe® Interface Logic
 - DDR Memory Interface Controllers
 - XDMA (Xilinx DMA) Logic
 - Hardware Configuration
 - Lifecycle Management
- Server / Workstation:
 - Operating System
 - Runtime library (XRT)
 - Runtime drivers (XRT)
 - Firmware & Boot Loader



(2) FPGA Acceleration Cards: AMD Vitis Design Flow

- 1 Profile Applications and Identify Performance-critical Functions



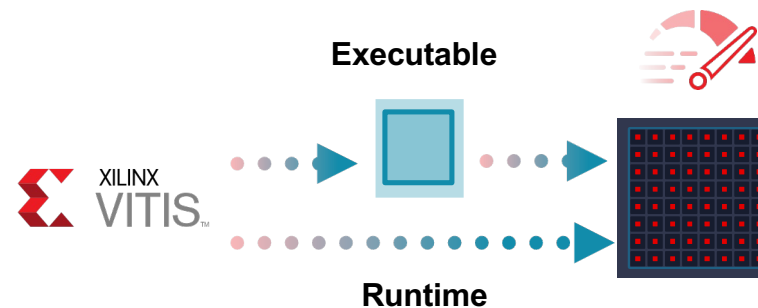
- 3 Build, Analyze & Debug : Validate Performance Goals Met



- 2 Design Accelerated Kernels

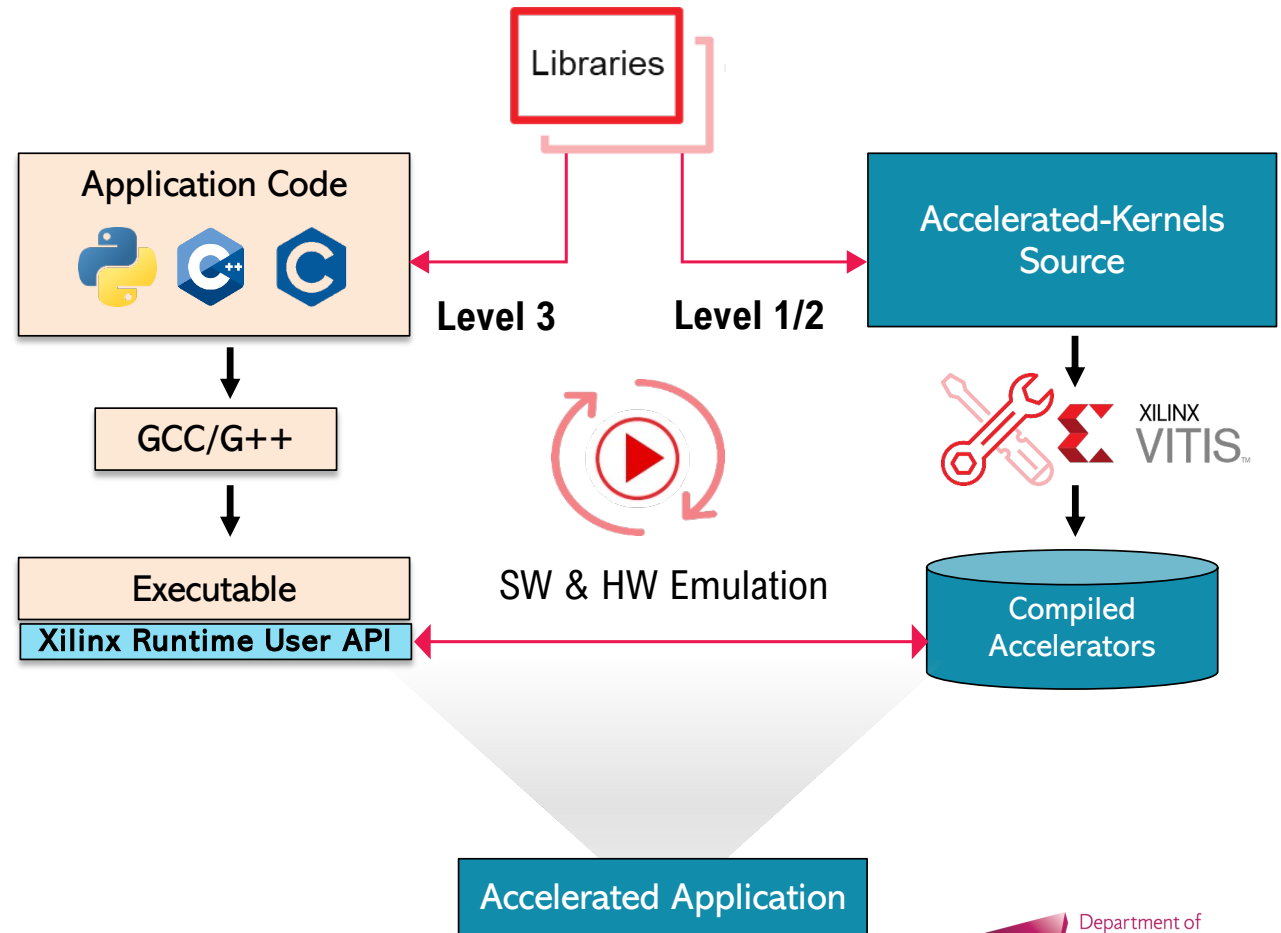


- 4 Deploy Accelerated Application on Xilinx Platforms

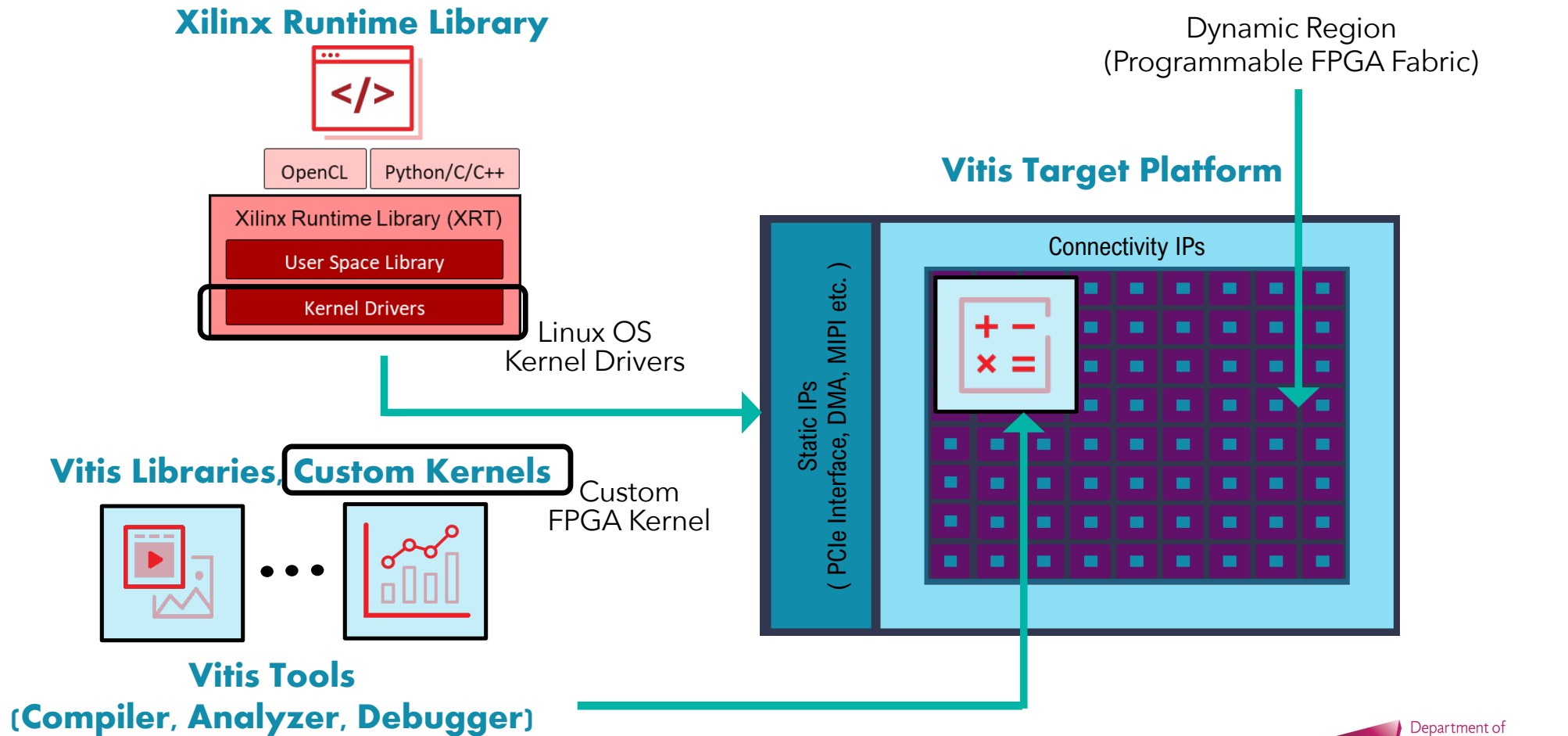


(2) FPGA Acceleration Cards: Server Applications

- Vitis Level 1 Abstraction
 - Kernel Level Acceleration
 - H/W Developers / HLS Experts
- Vitis Level 2 Abstraction
 - Platform Level Acceleration
 - Advanced S/W & H/W Users
- Vitis Level 3 Abstraction
 - Library Based Acceleration
 - Application Developers
 - Languages: Python, C++

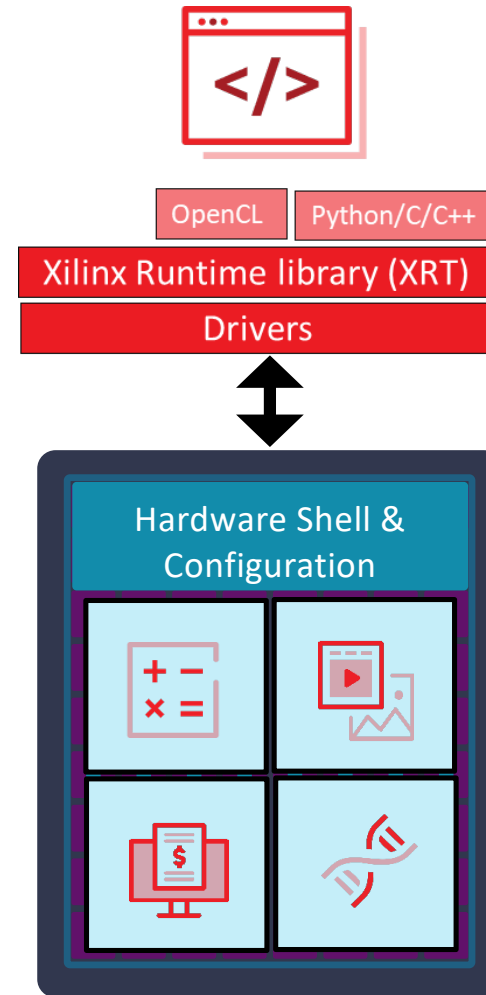


(2) FPGA Acceleration Cards: Server Applications

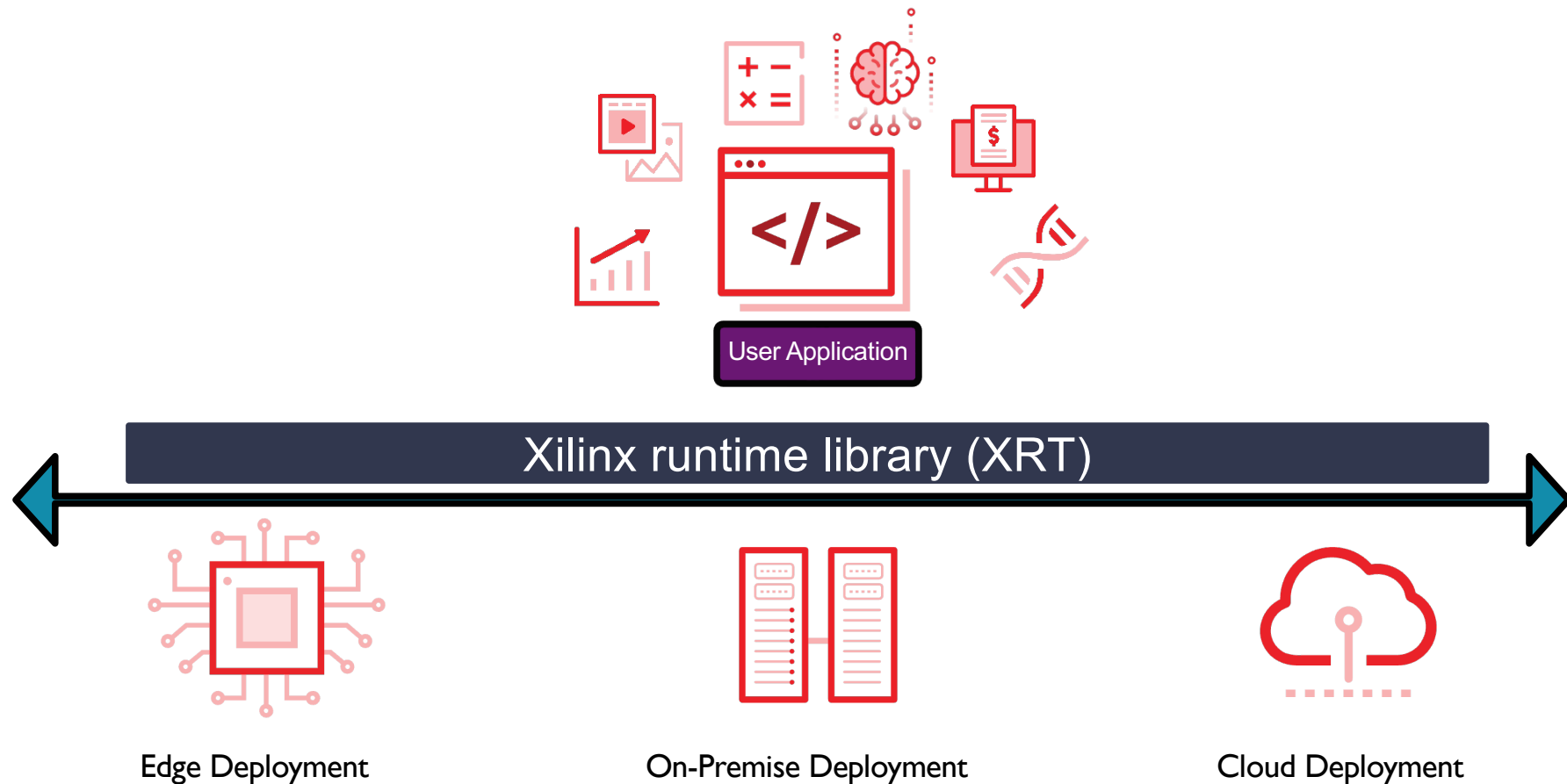


(2) FPGA Acceleration Cards: Xilinx Runtime Library (XRT)

- Standardized, Open-Source Software Interface
 - Between Host Application & Accelerators on Device
 - Use Low-Level APIs or OpenCL, Python Bindings
- Responsible for:
 - FPGA Bitstream Download
 - Accelerator Binaries
 - Memory Management
 - Data Movement between Host and Device
 - Execution Management
 - Trigger, Sequence & Synchronize Computations
 - Board Management
 - Board Recovery, Power Measurement etc.

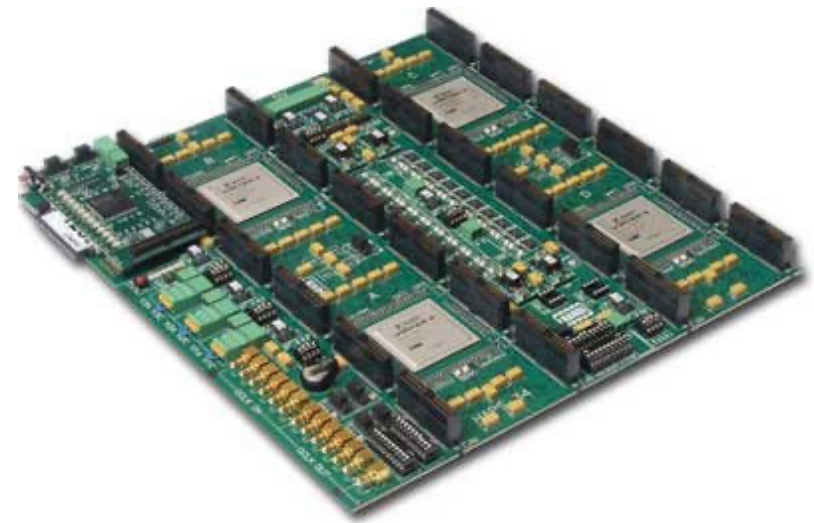


(2) FPGA Acceleration Cards: Xilinx Runtime Library (XRT)



(3) ASIC Prototyping with FPGA

- Pre-Silicon Simulation (Synopsys VCS, Cadence NCSim)
 - **SIMULATION** Speed Slow with Billions of Gates
 - Verification Coverage Limited
 - Software/Hardware **CO-SIMULATION** Difficult
- FPGA-based ASIC Prototyping
 - (Real-time) **HARDWARE VALIDATION**
 - Support SW+HW **SYSTEM TESTS**
 - Speed may be slower than final design
 - Form factor can be much larger than actual design



(4) FPGA Emulation Platforms

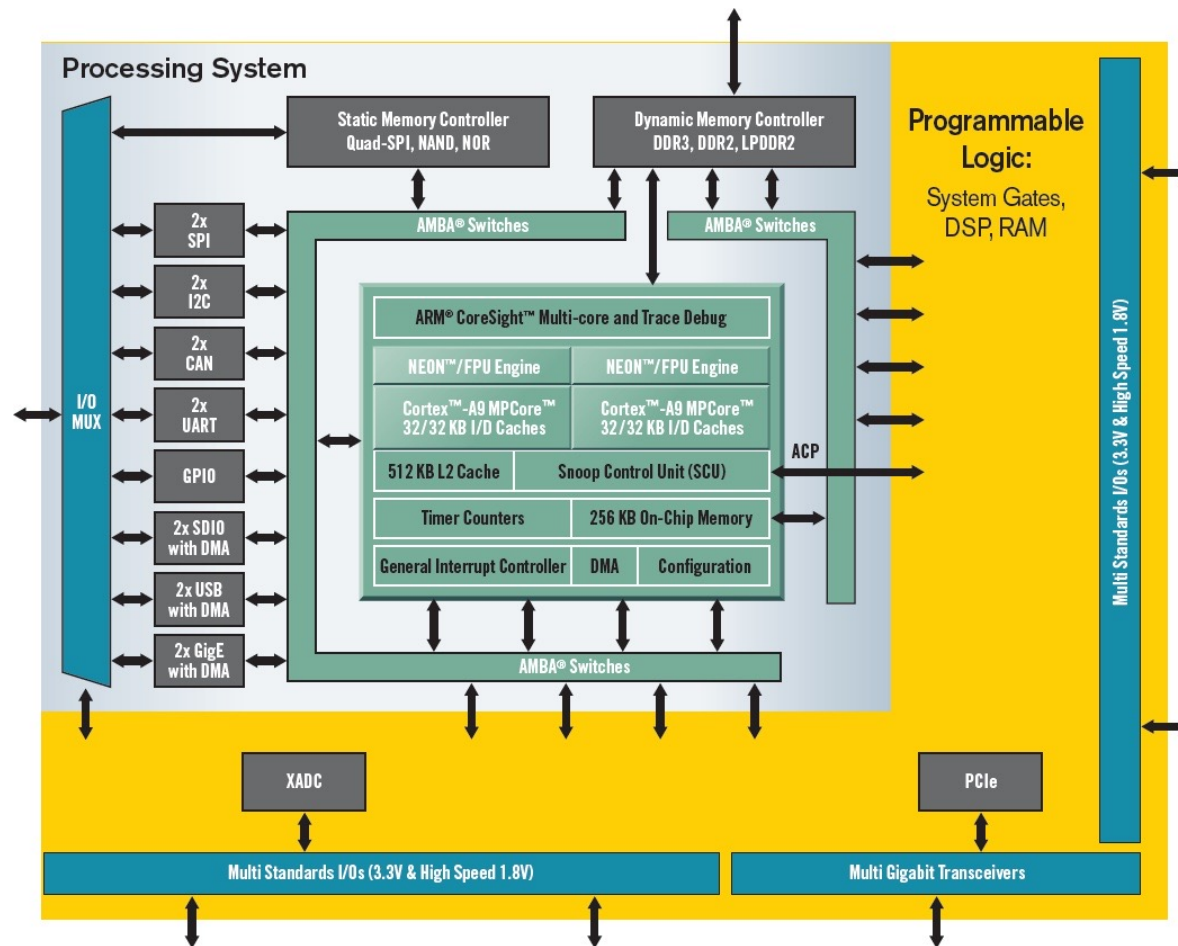
- Pre-Silicon ASIC **SIMULATION**
 - FPGA-assisted Verification
 - 100s of millions of lines of code
 - >60B gates
 - Can be used in **MANY ASIC VERIFICATION**
- Examples:
 - Synopsys Zebu-200 Emulation System
 - Cadence Palladium Emulation System



Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

MPSoC: Soft MCU Cores + Hard MCU Cores + FPGA



Soft MCU: Building Blocks (1 of 3)

- Central Processing Unit (CPU)
 - CPU is the fundamental block responsible for program execution.
 - It is a state machine tasked to execute commands in a sequence of instructions.
- Arithmetic Logic Unit (ALU)
 - ALU is a sub-part of CPU responsible for executing arithmetic and logic operations.
 - ALU is usually composed of combinational circuits.
 - ALU can perform addition, subtraction, multiplication, division and different types of shift and logical operations.

Soft MCU: Building Blocks (2 of 3)

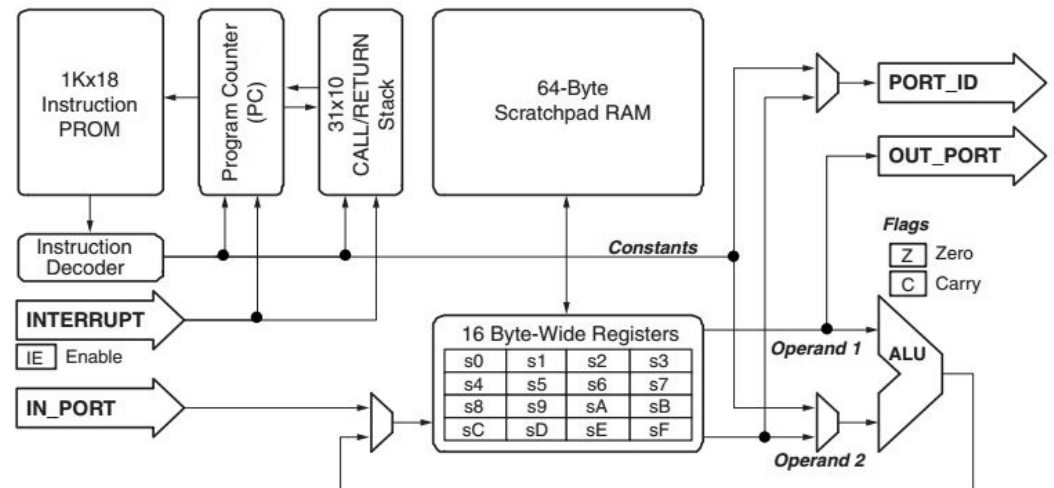
- Memory
 - Program memory resides in ROM (Flash).
 - Data memory resides in RAM.
 - MCU should have both ROM and RAM blocks.
- Oscillator / Clock
 - MCU needs a clock signal to perform sequential logics.
 - Clock is generated either by externally or internally.
 - Previous microcontrollers had only one clock source.
 - New microcontrollers may have multiple clock sources for different blocks.

Soft MCU: Building Blocks (3 of 3)

- GPIO
 - A microcontroller interacts with the outside world through its input and output pins.
 - These are called general purpose input and output (GPIO).
 - Most of the times, GPIO pins operate with digital data.
 - These pins may be of use for other applications as well such as digital communication and analog to digital conversion.
- Other Blocks
 - A microcontroller may also have other blocks generally called as peripherals.
 - Digital communication, analog to digital converter (ADC), and digital to analog converter (DAC) can be counted as such blocks.

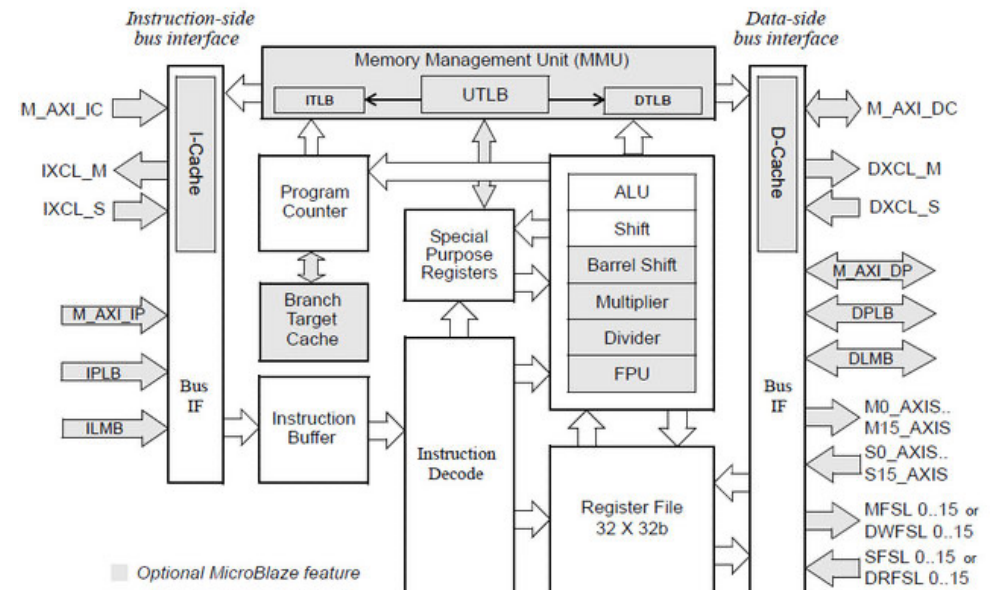
Soft MCU Example: PicoBlaze MCU

- PicoBlaze
 - 8-bit soft-core microcontroller developed by AMD Xilinx.
- This **SIMPLE MCU** core can run on many different AMD Xilinx FPGA Devices:
 - Versal Series (High-end)
 - Virtex Series (High-end)
 - Zynq Series (Mid-to-High-End)
 - Kintex Series (Mid-End)
 - Artix Series (Low-to-Mid-End)
 - Spartan Series (Low-end)



Soft MCU Example: MicroBlaze MCU

- MicroBlaze
 - 32-bit soft-core microcontroller developed by AMD Xilinx.
 - It has a much more complex architecture compared to PicoBlaze.
- MicroBlaze can also run on all AMD Xilinx FPGA Devices, provided these devices have sufficient resources.



Hard MCU Example: ARM Cortex A9 Overview

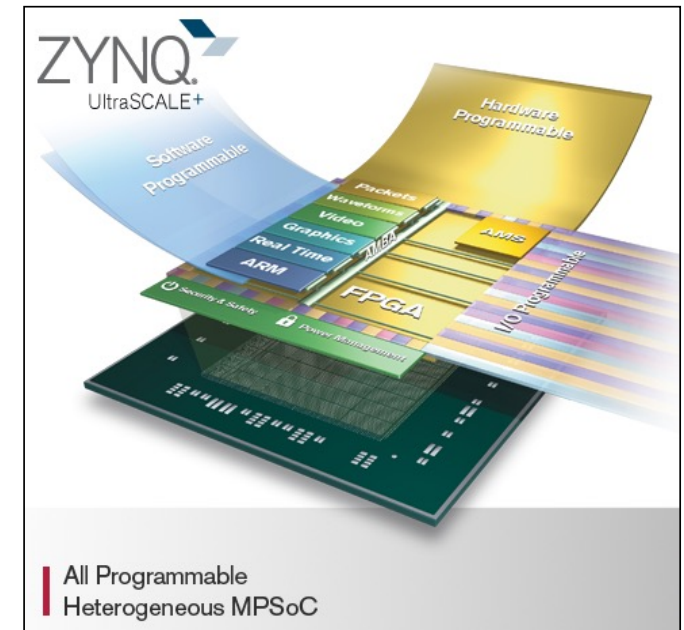
- Dual-core processor cluster
- 2.5 DMIP/MHz per processor (DMIP = Dhrystone MIPS)
- Harvard architecture
- Self-contained 32KB L1 caches for instructions and data
- 512KB on-chip unified L2 cache
- Automatic cache coherency between processor cores
- 1GHz operation (fastest speed grade)

Hard MCU Example: ARM Cortex A9 Microarchitecture

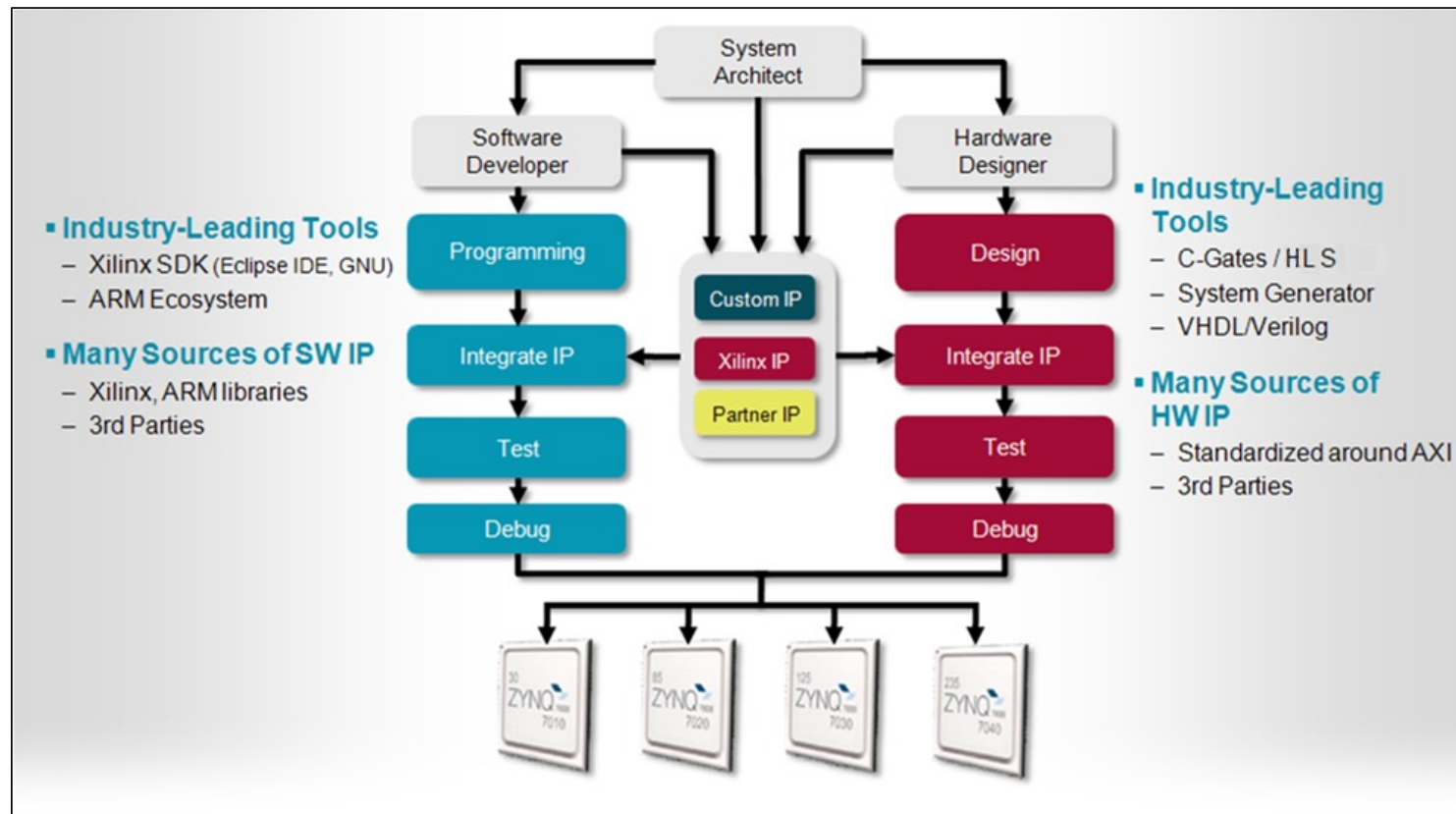
- Variable length, out-of-order, eight-stage, super-scalar instruction pipeline
 - Advanced pre-fetch with parallel branch pipeline
 - Early branch prediction and resolution
 - Multi-issued into different pipelines:
 - Primary data processing pipeline
 - Secondary full data processing pipeline
 - Load-store pipeline
 - Compute engine (FPU/NEON) pipeline
- Speculative execution
 - Supports virtual renaming of ARM physical registers to remove pipeline stalls
 - Increased processor utilization and hiding of memory latencies
 - Increased performance by hardware unrolling of code loops
 - Reduced interrupt latency via speculative entry to Interrupt Service Routine

MPSoC Examples: Zynq-7000 vs Zynq UltraScale+

Feature	Zynq-7000	Zynq UltraScale+
Target Market Segments	Cost-sensitive, moderate-performance embedded systems	High-performance embedded AI, vision, SDR, real-time control
ARM Cortex CPU cores	32-bit dual-core A9	64-bit quad-core A53 + dual-core R5
FPGA Fabric	28nm Artix / Kintex	16nm UltraScale+
Multimedia / AI Ready	NO	YES

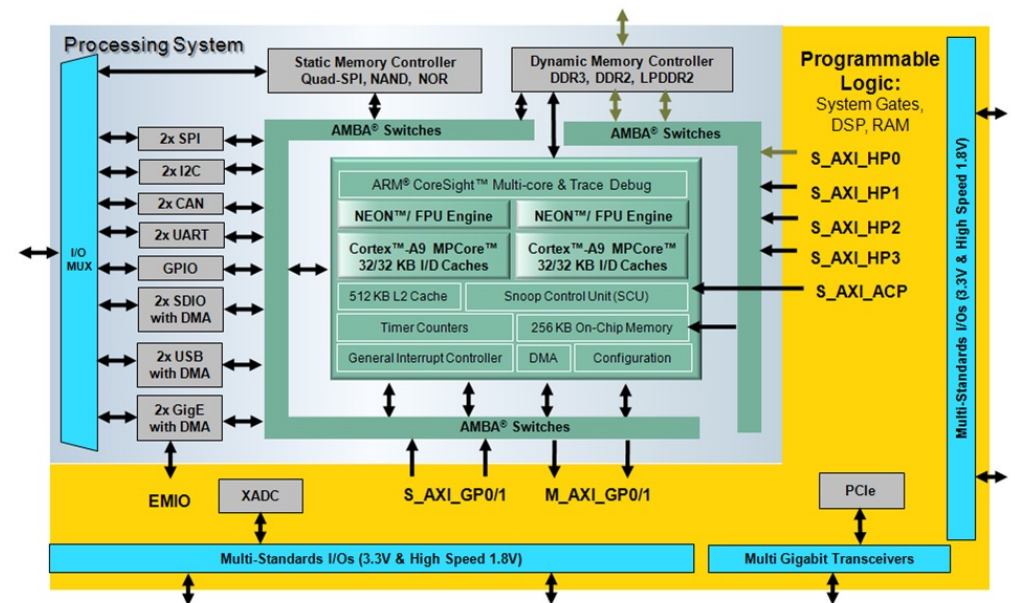


MPSoC Examples: AMD Xilinx Zynq System Design Flow



MPSoC Examples: Zynq-7000 Family Highlights

- Complete ARM®-based Processing System (PS)
 - Application Processor Unit (APU)
 - Dual ARM Cortex™-A9 processor
 - Caches and support blocks
 - Fully integrated memory controllers
 - I/O peripherals
- Tightly Integrated Programmable Logic (PL)
 - Used to extend the processing system
 - Scalable density and performance
- Flexible Array of I/O
 - Wide range of external multi-standard I/O
 - High-performance integrated serial transceivers
 - Analog-to-digital converter inputs



MPSoC Examples: Zynq-7000 Family

- Embedded design with Zynq is based on:
 - Processor and peripherals
 - Dual ARM® Cortex™ -A9 processors of Zynq-7000 SoC
 - AXI interconnect
 - AXI component peripherals
 - Reset, clocking, debug ports
 - Software platform for processing system
 - Bare Metal Applications or OS's (e.g. Linux, FreeRTOS)
 - C language support
 - Processor services
 - C drivers for hardware
 - User application
 - Interrupt service routines (optional)

Zynq-7000 Data Sheets: Overview

- For details, please refer to the corresponding datasheet.
- As an engineer, it is important to learn how to read this kind of datasheets.
- For example, how to order a correct Part Number (P/N)?

XILINX		Zynq-7000 SoC Data Sheet: Overview
DS190 (v1.11.1) July 2, 2018		Product Specification
Zynq-7000 SoC First Generation Architecture		
The Zynq-7000 family is based on the Xilinx SoC architecture. These products integrate a feature-rich dual-core or single-core ARM® Cortex™-A9 based processing system (PS) and 28 nm Xilinx programmable logic (PL) in a single device. The ARM Cortex-A9 CPUs are the heart of the PS and also include on-chip memory, external memory interfaces, and a rich set of peripheral connectivity interfaces.		
Processing System (PS)		
ARM Cortex-A9 Based Application Processor Unit (APU)		I/O Peripherals and Interfaces
• 2.5 DMIPS/MHz per CPU • CPU frequency: up to 1 GHz • Coherent multiprocessor support • ARMv7-A architecture • TrustZone® security • ThumbEE-2 instruction set • Jazelle® RCT execution Environment Architecture • NEON™ media-processing engine • Single and double precision Vector Floating Point Unit (VFPv3) • CoreSight™ and Program Trace Macrocell (PTM) • Timer and Interrupts • Three watchdog timers • One global timer • Two triple-timer counters		• Two 10/100/1000 tri-speed Ethernet MAC peripherals with IEEE Std 802.3 and IEEE Std 1588 revision 2.0 support • Scatter-gather DMA capability • Recognition of 1588 rev. 2 PTP frames • GMII, RGMII, and SGMII interfaces • Two USB 2.0 OTG peripherals, each supporting up to 12 Endpoints • USB 2.0 compliant device IP core • Supports on-the-go, high-speed, full-speed, and low-speed modes • Intel ENCI compliant USB host • 8-bit LULPI external PHY interface • Two full CAN 2.0B compliant CAN bus interfaces • CAN 2.0-A and CAN 2.0-B and ISO 11898-1 standard compliant • External PHY interface • Two SD/SDIO 2.0/MMC3.31 compliant controllers • Two full-duplex SPI ports with three peripheral chip selects • Two high-speed UARTs (up to 1 Mb/s) • Two master and slave I2C interfaces • GPIO with four 32-bit banks, of which up to 54 bits can be used with the PS I/O (one bank of 32b and one bank of 22b) and up to 64 bits (up to two banks of 32b) connected to the Programmable Logic • Up to 54 flexible multiplexed I/O (MIO) for peripheral pin assignments
Caches		Interconnect
• 32 KB Level 1 4-way set-associative instruction and data caches (independent for each CPU) • 512 KB 8-way set-associative Level 2 cache (shared between the CPUs) • Byte-parity support		• High-bandwidth connectivity within PS and between PS and PL • ARM AMBA® AXI based • QoS support on critical masters for latency and bandwidth control
On-Chip Memory		
• On-chip boot ROM • 256 KB on-chip RAM (OCM) • Byte-parity support		
External Memory Interfaces		
• Multiprotocol dynamic memory controller • 16-bit or 32-bit interfaces to DDR3, DDR3L, DDR2, or LPDDR2 memories • ECC support in 16-bit mode • 1GB of address space using single rank of 8-, 16-, or 32-bit-wide memories • Static memory interfaces • 8-bit SRAM data bus with up to 64 MB support • Parallel NOR flash support • CNF11.0 NAND flash support (1-bit ECC) • 1-bit SPI, 2-bit SPI, 4-bit SPI (quad-SPI), or two quad-SPI (8-bit) serial NOR flash		
8-Channel DMA Controller		
• Memory-to-memory, memory-to-peripheral, peripheral-to-memory, and scatter-gather transaction support		
© Copyright 2012–2018 Xilinx, Inc., Xilinx, the Xilinx logo, Artix, iSE, Kintex, Spartan, Virtex, Verso, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. AMBA, AMBA Designer, ARM, ARM Cortex-A9, CoreSight, Cortex, and PrimeCell are trademarks of ARM in the EU and other countries. PCI, PCIe, and PCI Express are trademarks of PCI-SIG and used under license. All other trademarks are the property of their respective owners.		
DS190 (v1.11.1) July 2, 2018		www.xilinx.com
Product Specification		1

Zynq-7000 Data Sheets: Ordering Information

- Each Letter has a special meaning:
 - XC7Z007S**-1CLG484C (Commercial)
 - XC7Z007S**-2CLG484E (Extended)
 - XC7Z007S**-2CLG484I (Industrial)
 - XC7Z007S**-2CLG484C (Commercial)

Device	Speed Grade and Junction Temperature Range		
	Commercial (C) 0°C to +85°C	Extended (E) 0°C to +100°C	Industrial (I) -40°C to +100°C
XC7Z007S XC7Z012S XC7Z014S	-1	-2	-1, -2
XC7Z010 XC7Z015 XC7Z020	-1	-2, -3	-1, -2, -1L
XC7Z030 XC7Z035 XC7Z045	-1	-2, -3	-1, -2, -2L
XC7Z100	-1	-2	-1, -2, -2L

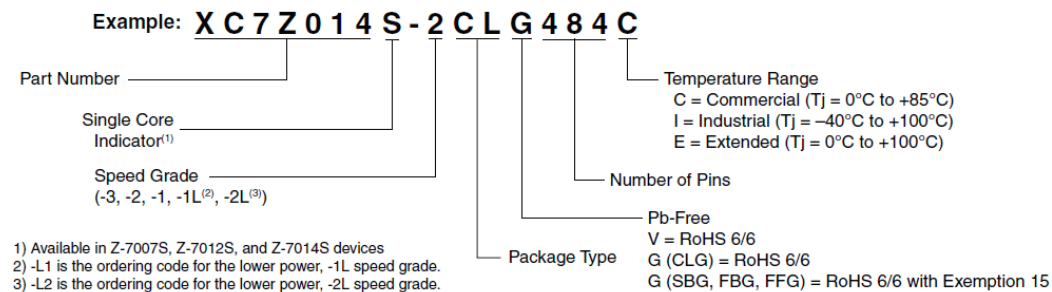


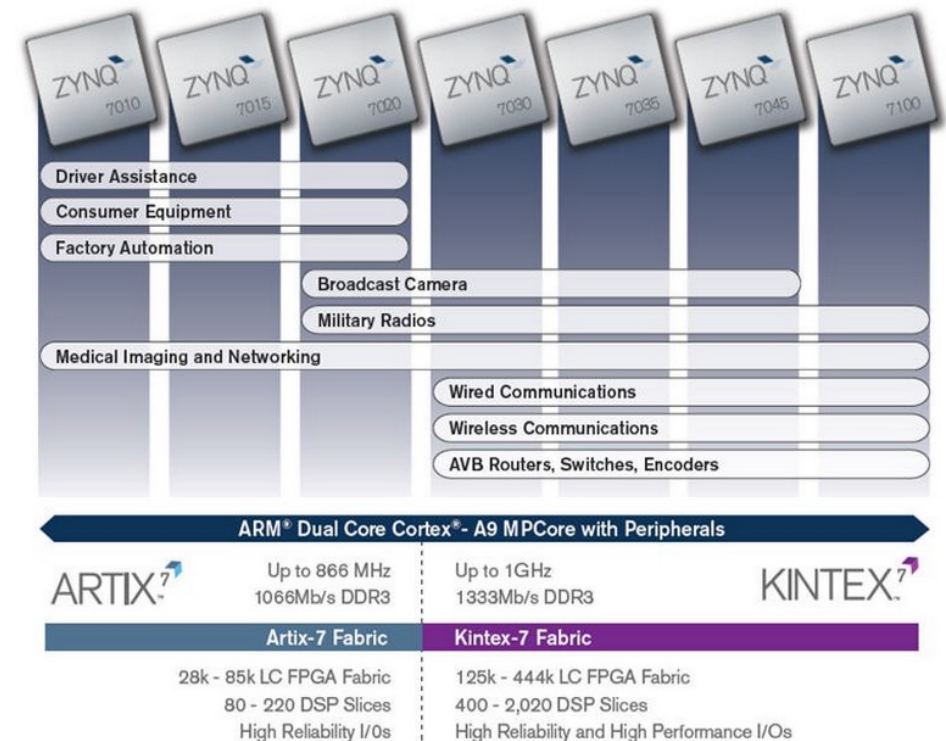
Figure 4: Ordering Information

Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

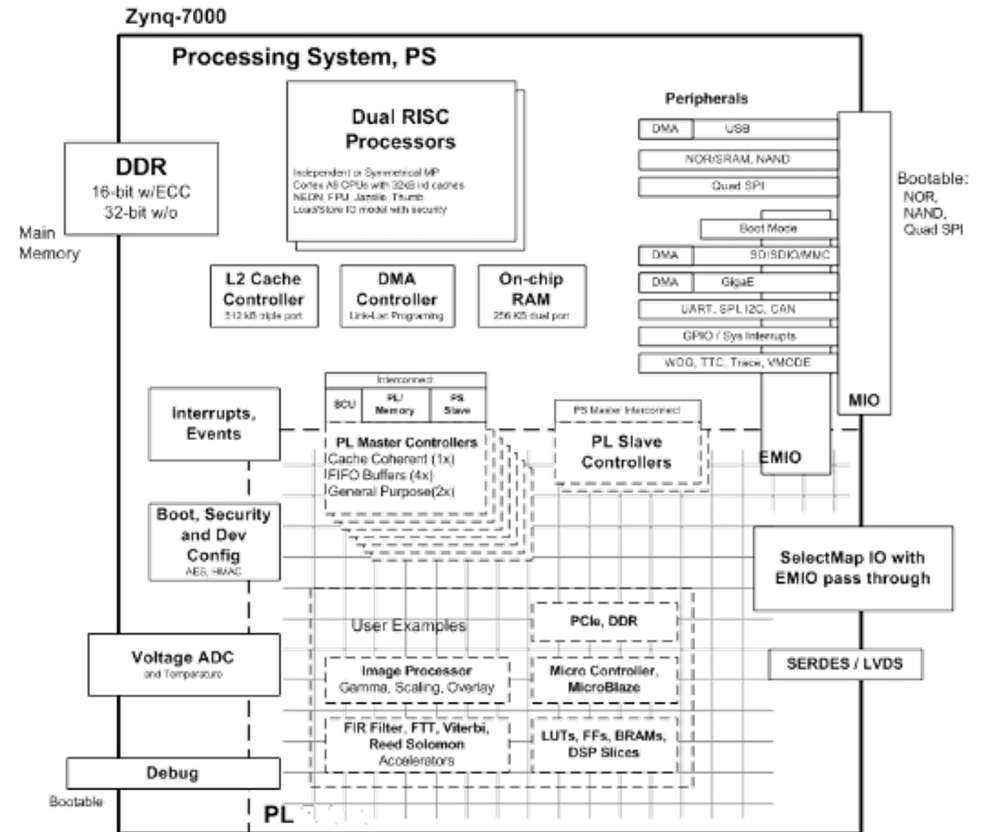
Zynq-7000 SoC Architectural Overview

- Zynq-7000 SoC has 2 major sections (PS & PL)
- **PROCESSING SYSTEM (PS)**
 - Dual ARM Cortex-A9 processor based
 - Single core versions available
 - Multiple peripherals
 - Hard silicon core
- **PROGRAMMABLE LOGIC (PL)**
 - Uses the same 7 series programmable logic
 - Artix™-based devices: Z-7010, Z-7015 and Z-7020 (high-range I/O banks only)
 - Kintex™-based devices: Z-7030, Z-7035, Z-7045, and Z-7100 (mix of high-range and high-performance I/O banks)



Zynq Processing System (PS): Components

- Zynq Processing System (PS) consists of:
 - Application processing unit (APU)
 - I/O peripherals (IOP)
 - Multiplexed I/O (MIO), extended multiplexed I/O (EMIO)
 - Memory interfaces
 - PS interconnect
 - DMA, Timers
 - Public and private
 - General interrupt controller (GIC)
 - On-chip memory (OCM): ROM and RAM
 - Debug controller: CoreSight



Zynq Processing System (PS): Memory Resources

- On-chip memory (OCM)
 - RAM
 - Boot ROM
- DDRx dynamic memory controller supporting:
 - LPDDR2
 - DDR2
 - DDR3
- Flash/static, memory controller supporting:
 - SRAM
 - QSPI
 - NAND/NOR FLASH

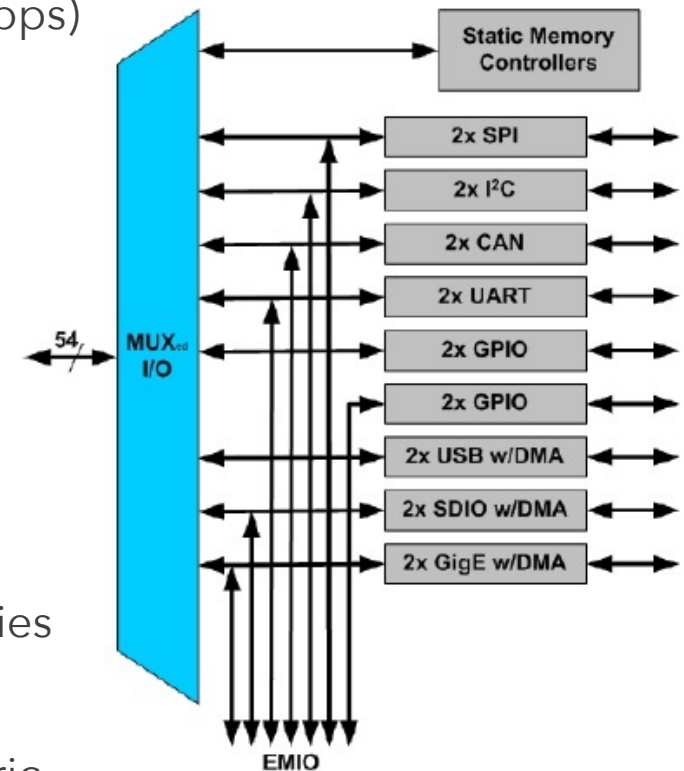
Zynq Processing System (PS): Memory Map

- The Cortex-A9 processors in the Zynq Processing System (PS) use a **32-BIT ADDRESS SPACE**.
- All PS peripherals and AXI-connected PL peripherals are memory mapped into this address space.
- AXI slaves in PL are accessible from PS via the General-Purpose (GP) AXI master ports:
 - Address range 4000_0000 to 7FFF_FFFF is mapped to AXI GP0
 - Address range 8000_0000 to BFFF_FFFF is mapped to AXI GP1

FFFC_0000 to FFFF_FFFF	OCM
FD00_0000 to FFFB_FFFF	Reserved
FC00_0000 to FCFF_FFFF	Quad SPI linear address
F8F0_3000 to FBFF_FFFF	Reserved
F890_0000 to F8F0_2FFF	CPU Private registers
F801_0000 to F88F_FFFF	Reserved
F800_1000 to F880_FFFF	PS System registers,
F800_0C00 to F800_0FFF	Reserved
F800_0000 to F800_0BFF	SLCR Registers
E600_0000 to F7FF_FFFF	Reserved
E100_0000 to E5FF_FFFF	SMC Memory
E030_0000 to E0FF_FFFF	Reserved
E000_0000 to E02F_FFFF	IO Peripherals
C000_0000 to DFFF_FFFF	Reserved
8000_0000 to BFFF_FFFF	PL (MAXI_GP1)
4000_0000 to 7FFF_FFFF	PL (MAXI_GP0)
0010_0000 to 3FFF_FFFF	DDR(address not filtered by SCU)
0004_0000 to 000F_FFFF	DDR(address filtered by SCU)
0000_0000 to 0003_FFFF	OCM

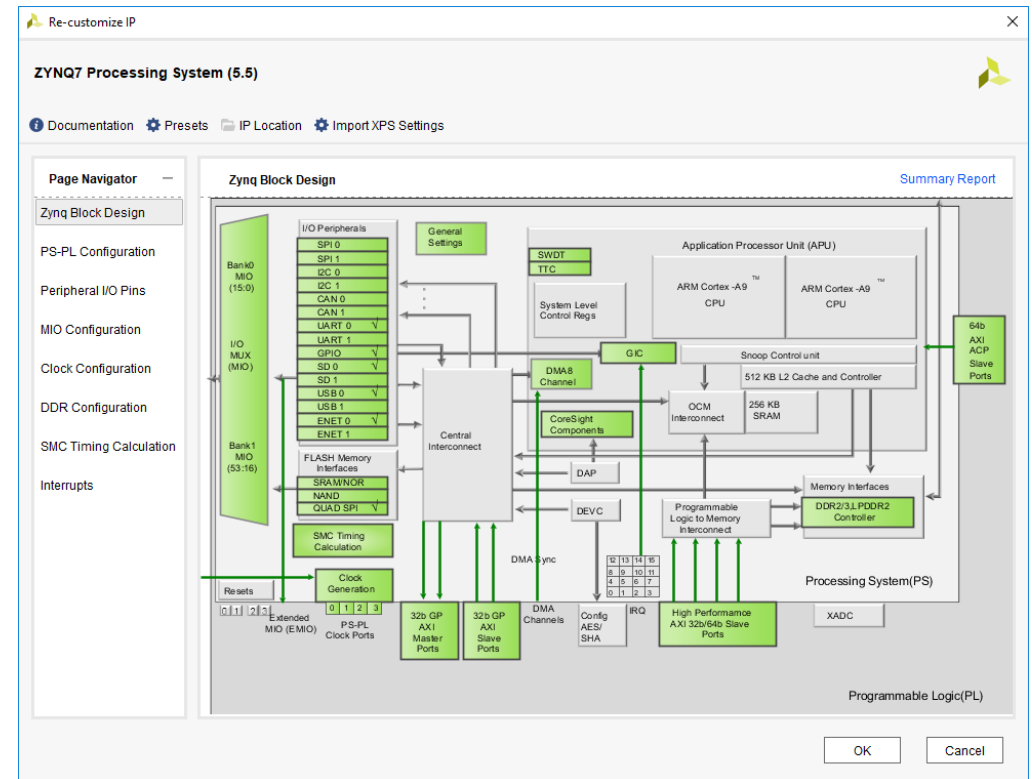
Zynq Processing System (PS): Built-in Peripherals

- 2 x SD/SDIO interfaces (compatible with memory, I/O, combo SD cards)
- 2 x Tri-Mode GigE (supporting 10Mbps, 100Mbps, or 1000Mbps)
- 2 x USB 2.0 OTG (supporting both host and device modes)
- 2 x CAN 2.0B interfaces
- Multiple SPI, I2C, UART controllers
- 4 x 32-bit GPIO Blocks
 - 54 GPIOs via MIO, more GPIOs via EMIO
- Multiplexed Input / Output (MIO)
 - Routes PS peripheral signals to dedicated pins
 - Supports pin multiplexing of peripherals & static memories
- Extended Multiplexed Input / Output (EMIO)
 - Allows internal routing of PS peripheral signals to PL fabric



Zynq Processing System (PS): Customization Overview

- Zynq Block Design
- PS-PL Interface Configuration
- Peripheral I/O Pins
- MIO Configuration/Table View
- Clock Configuration
- DDR Configuration (e.g. DDR Clock)
- SMC Timing Calculation (e.g. NOR Flash)
- Interrupts



Zynq Processing System (PS): Customization Pages

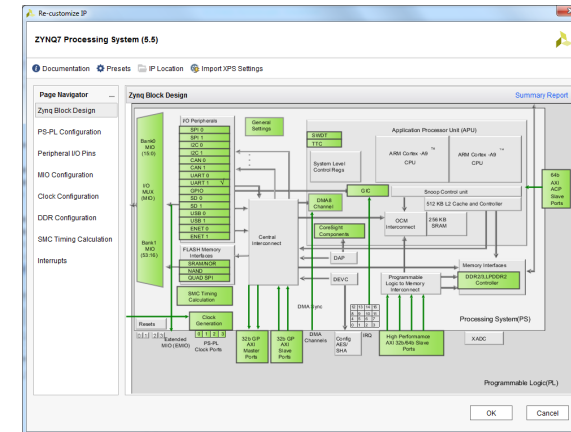
- Provides a graphical view of the PS to configure:

1 or 2 cores, enable NEON / FPU?

- ARM cores
- I/O peripherals
- DDR controller
- Memory systems

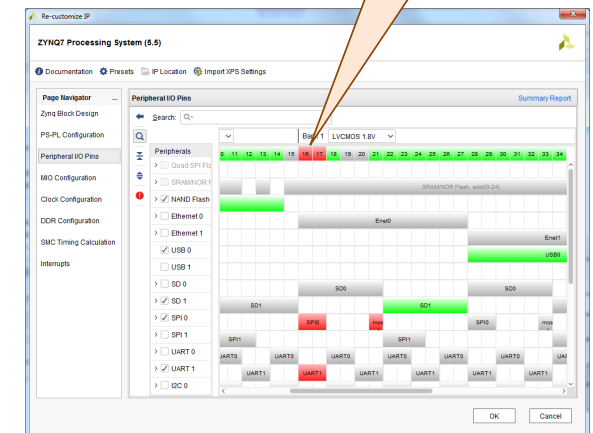
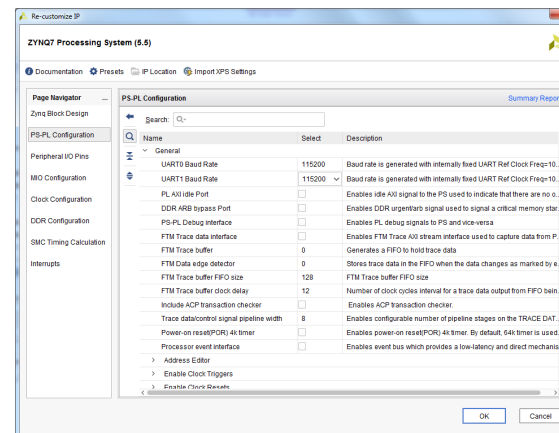
Enable peripherals, pin assignments

Specify DDR, memory sizes, timings



Check for Conflicts

- I/O partitioning between dedicated PS pins and programmable logic I/O
- PS is configured via a set of memory-mapped configuration registers

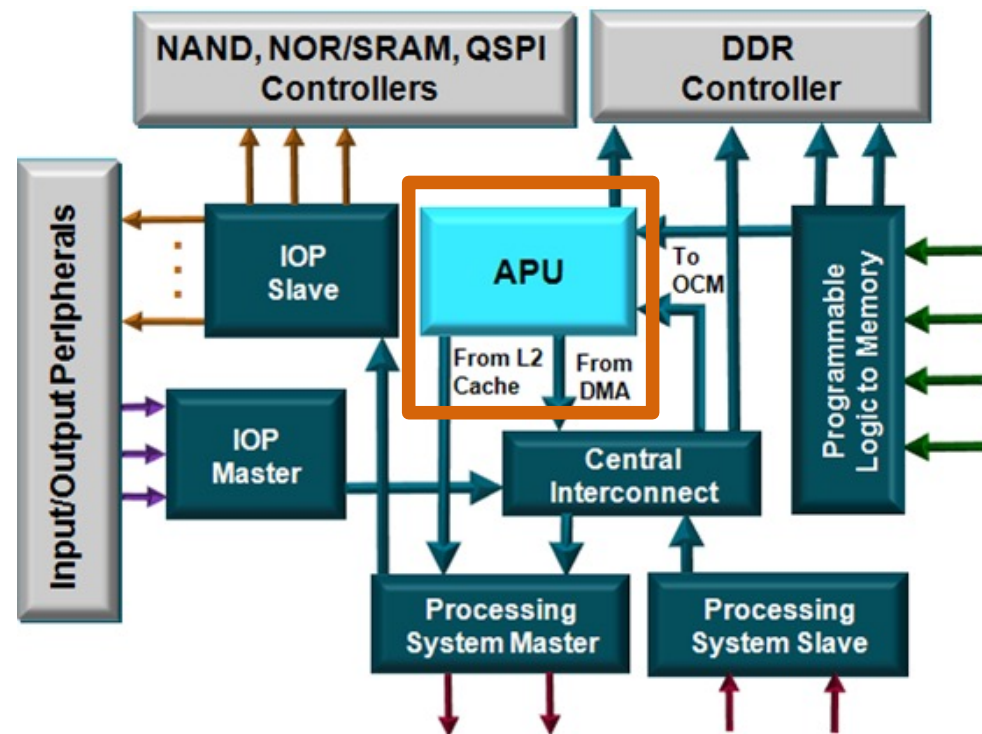


Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

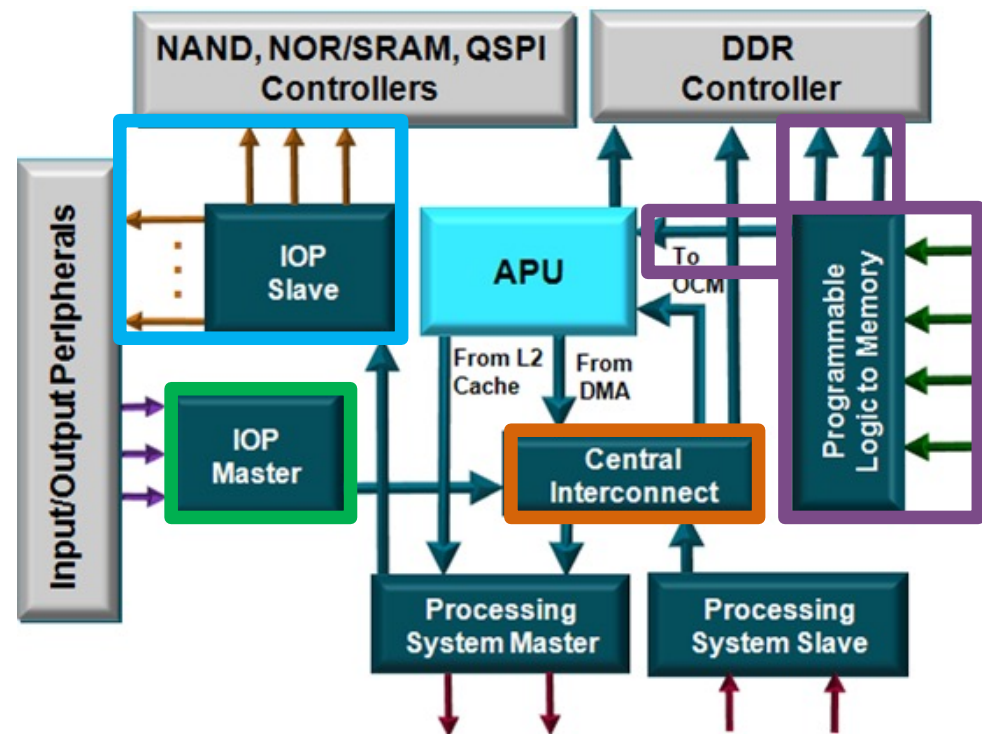
Zynq-7000 Processing System Block Diagram (1 of 3)

- Zynq Application Processing Unit (APU)
 - 2 ARM Cortex-A9 Cores
 - Shared L2 Unified Cache (512KB)
- ARM Cortex-A9 Core
 - ARMv7-A Architecture
 - 32KB L1 Instruction Cache
 - 32KB L1 Data Cache
 - MMU
 - NEON SIMD Engine
 - VFPv3 FPU
 - Generic Interrupt Controller
 - Global Timer / Watchdog Timer
 - CoreSight Debug (JTAG, trace)



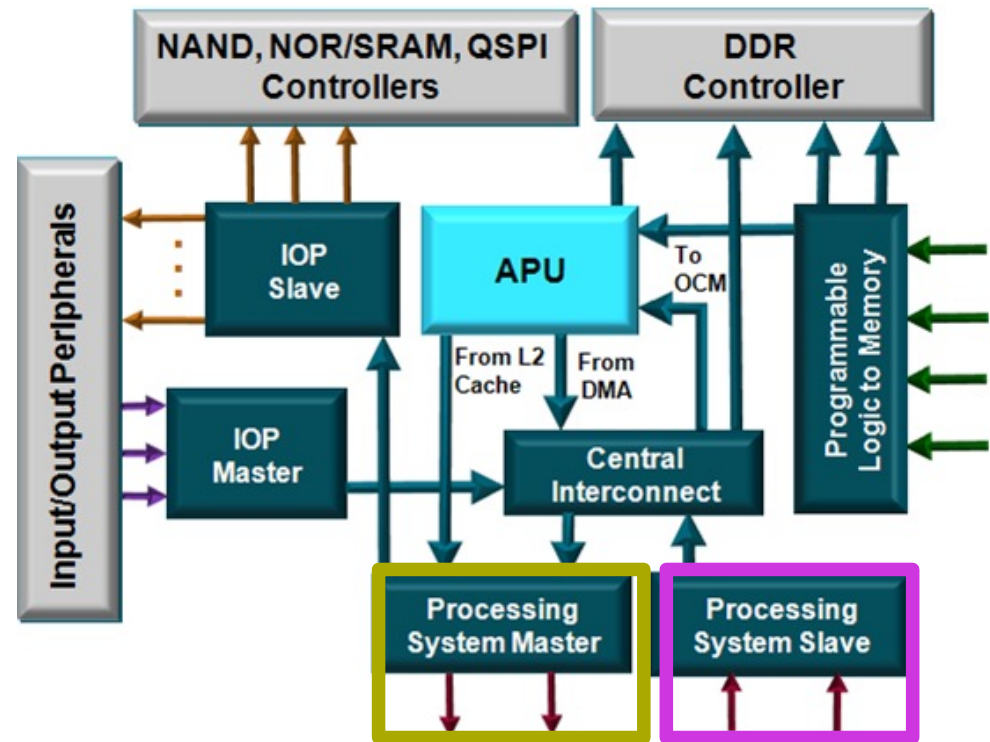
Zynq-7000 Processing System Block Diagram (2 of 3)

- Programmable Logic (PL) to Memory
 - Two ports to DDR
 - One port to OCM SRAM
- Central Interconnect
 - Enables other interconnects to communicate
- Peripheral Master
 - USB, GigE, SDIO connects to DDR and PL via the central interconnect
- Peripheral Slave
 - CPU, DMA, and PL access to IOP peripherals



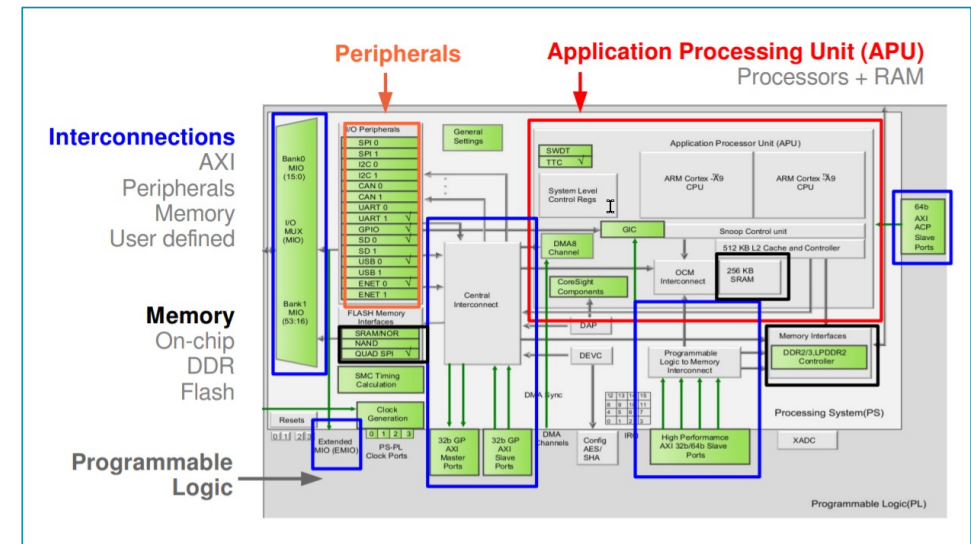
Zynq-7000 Processing System Block Diagram (3 of 3)

- Processing System Master
 - Two ports from the PS to PL
 - Connects APU block to common peripherals through Central Interconnect
- Processing System Slave
 - Two ports from PL to PS

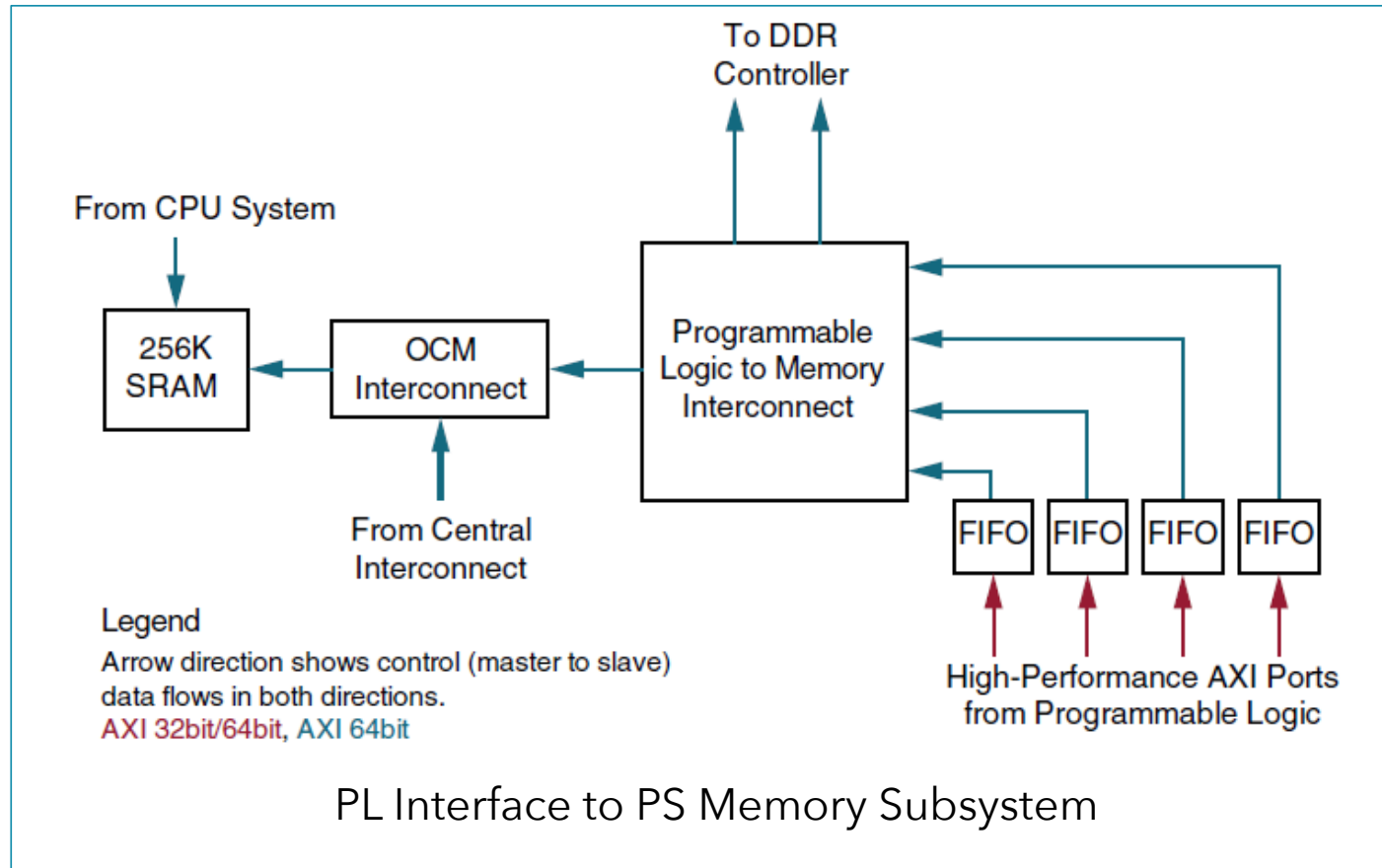


PS-PL Interfaces (1): 3 Types of AXI Interfaces

- AXI high-performance PS slave ports (HP0-HP3)
 - Configurable 32-bit or 64-bit data width
 - Access to OCM and DDR only
 - Sync to PS clock domain
 - **AXI FIFO INTERFACE (AFI)** are FIFOs (1KB) to smooth large data transfers
- AXI general-purpose ports (GP0-GP1)
 - Two **MASTERS** from PS to PL
 - Two **SLAVES** from PL to PS
 - 32-bit data width
 - Sync to processing system clock domain
- One 64-bit Accelerator Coherence Port (ACP) AXI Slave interface used by PL for **CACHE COHERENT ACCESS** to CPU MEMORY



PS-PL Interfaces (2): AXI Interface to Memory Subsystem



PS-PL Interfaces (3): Other Non-AXI Interfaces

- DMA, Interrupts, Events signals
 - Processor **EVENT BUS** for signaling event information to CPU
 - PL peripheral IP **INTERRUPTS** to PS General Interrupt Controller (GIC)
 - Four **DMA** channel RDY/ACK signals
- **EXTENDED MULTIPLEXED I/O** (EMIO) allows PS peripheral ports access to PL logic and device I/O pins
- Clock and Resets
 - Four PS **CLOCK** outputs to PL with Enable Control
 - Four PS **RESET** outputs to PL

Boot Sequence (1): PS boots first

- CPU0 boots from OCM ROM; CPU1 goes into a sleep state
- On-chip boot loader in OCM ROM (Stage 0 boot)
- Processor loads First Stage Boot Loader (FSBL) from external flash memory
 - NOR
 - NAND
 - Quad-SPI (QSPI)
 - SD Card
 - Load from JTAG; not a memory device—used for development/debug only
- Boot source selected via package bootstrapping pins
- Optional secure boot mode allows the loading of encrypted software from the flash boot memory

Boot Sequence (2): PS configures PL

- The Programmable Logic (PL) is configured after the PS boots
- Performed by application software accessing the hardware device configuration unit
 - Bitstream image transferred
 - 100-MHz, 32-bit Processor Configuration Access Port (PCAP) stream interface
 - Decryption/authentication hardware option for encrypted bitstreams
 - In secure boot mode, this option can be used for software memory load
 - Built-in DMA allows simultaneous PL configuration and OS memory loading

Boot Sequence (2): PCAP

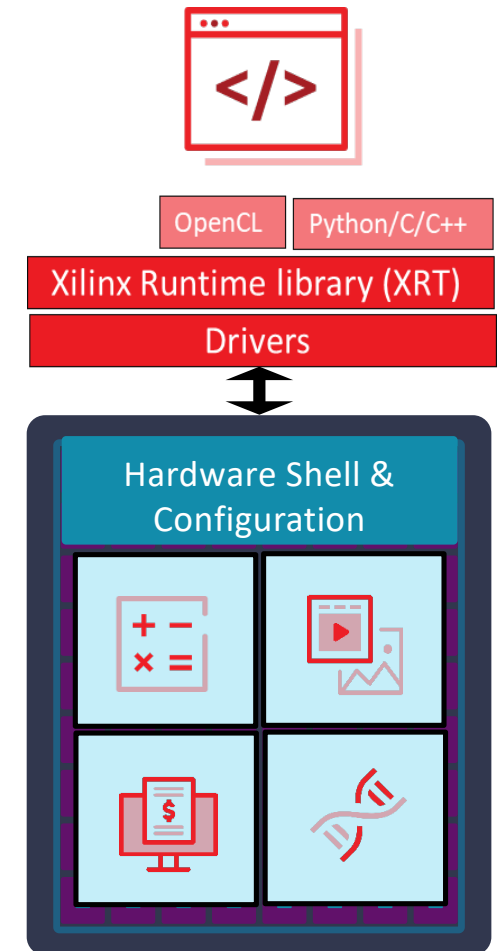
- What is PCAP?
 - Processor Configuration Access Port (PCAP) is an internal interface in Zynq devices that allows PS to configure PL.
- PCAP Features:
 - 32-bit bus, ~100MHz
 - Built-in DMA Engine
 - Supports Encrypted Bitstream and Runtime Reconfiguration
- When is PCAP used?
 - FSBL uses PCAP to load bitstream during **BOOT**.
 - Xilinx Runtime Library (**XRT**) on Linux can also use PCAP for dynamic PL reconfiguration at **RUNTIME**.

Boot Sequence (3): PS loads Linux Kernel

- After PL is configured, PS executes the Universal Bootloader (U-BOOT):
 - U-Boot loads the Linux Kernel Image into DDR RAM.
 - U-Boot loads the Device Tree into DDR RAM.
 - U-Boot loads the Root File System into DDR RAM (for RAM disk).
 - U-Boot may pass the Root File System info to the kernel (for external storage).
 - U-Boot launches the kernel to start the Linux OS.
- After that, Linux OS takes over:
 - The Linux OS mounts the Root File System (from RAM or external storage).
 - The Linux OS begins the User Space Processes.
 - User Applications use XRT to manage and control the PL FPGA fabric.

Revisit: Xilinx Runtime Library (XRT)

- What is XRT?
 - The Xilinx Runtime Library (XRT) serves as the core **MIDDLEWARE** and runtime layer for FPGA-based acceleration in both **EMBEDDED** and **SERVER PLATFORMS**.
- Purposes of XRT:
 - Abstract H/W control without dealing with AXI/PCAP
 - FPGA Bitstream & FPGA Kernel Management
 - FPGA Memory Buffer Management
 - FPGA Monitoring, Profiling, Debugging
 - Security and Board Management



Revisit: Xilinx Runtime Library (XRT)

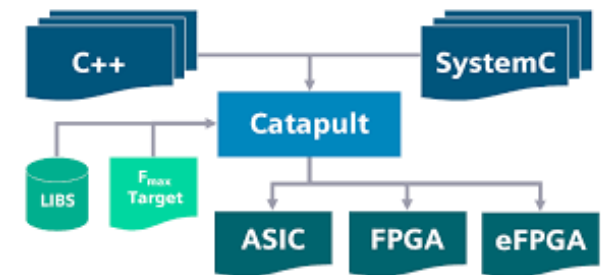
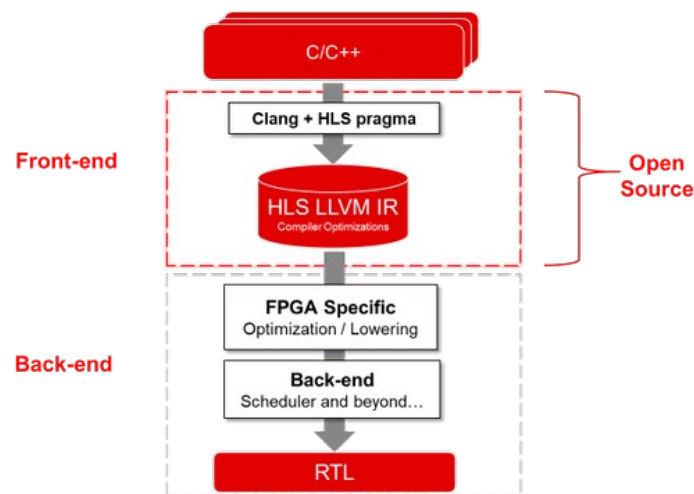
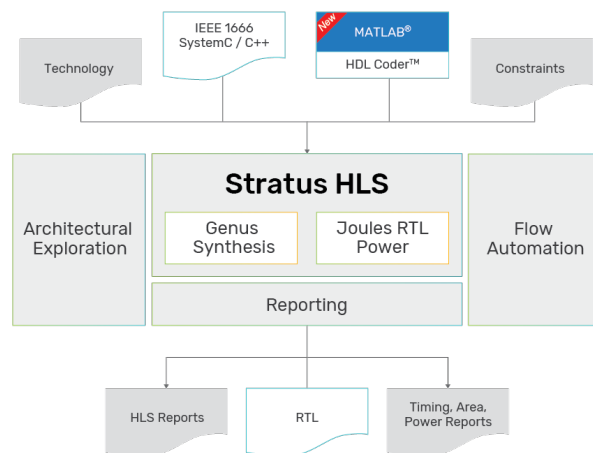
AMD FPGA Family	PS	PCIe	XRT on PS	XRT on Server
Spartan	NO	NO	NO	NO
Artix-7	NO	NO	NO	NO
Artix Ultrascale	NO	SOFT IP	NO	NO
Kintex Ultrascale+	NO	SOME P/N	NO	NO
Virtex Ultrascale+	NO	YES	NO	YES
Versal ACAP	YES	YES	YES	YES
Zynq Ultrascale+	YES	YES	YES	YES
Zynq-7000	YES	SOFT IP	YES	NO

Overview

- FPGA Use Cases
- Multi-Processor System-on-Chip (MPSoC)
- Zynq Processing System (PS) Customization
- PS Block Diagram, PS-PL Interfaces, Boot Sequence
- High Level Synthesis (HLS) in FPGA

High-Level Synthesis (HLS)

- High-level synthesis (HLS) is also known as C synthesis, electronic system-level (ESL) synthesis, algorithmic synthesis, or behavioral synthesis
- HLS is an automated design process, taking an **ABSTRACT BEHAVIORAL SPECIFICATION** of a digital system and realizing it in an **RTL STRUCTURE**.
 - Tools: Cadence Stratus HLS, Open Source LLVM-based HLS, Siemens / Mentor Graphics Catapult HLS, and **AMD / XILINX VITIS HLS**.



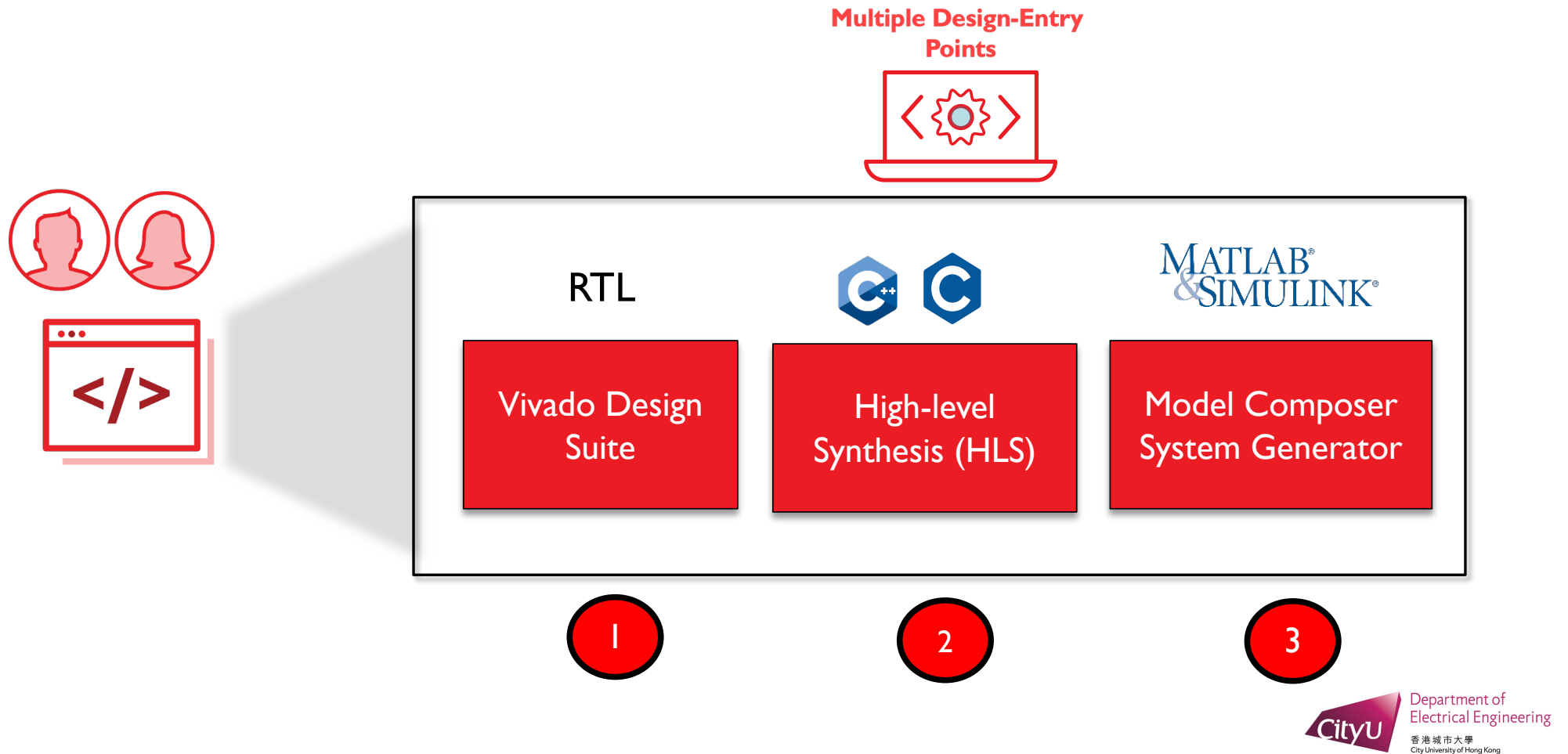
HLS: Develop in Familiar Programming Languages

- Source code for library functions written in C/C++
 - Some offer Python APIs directly callable in Python applications
- Automatically synthesized to RTL by Vitis Compilers
 - Using C/C++ to RTL High-level Synthesis (HLS) Technology
- No prior RTL design experience required
 - Libraries enable a familiar CPU/GPU-like design experience



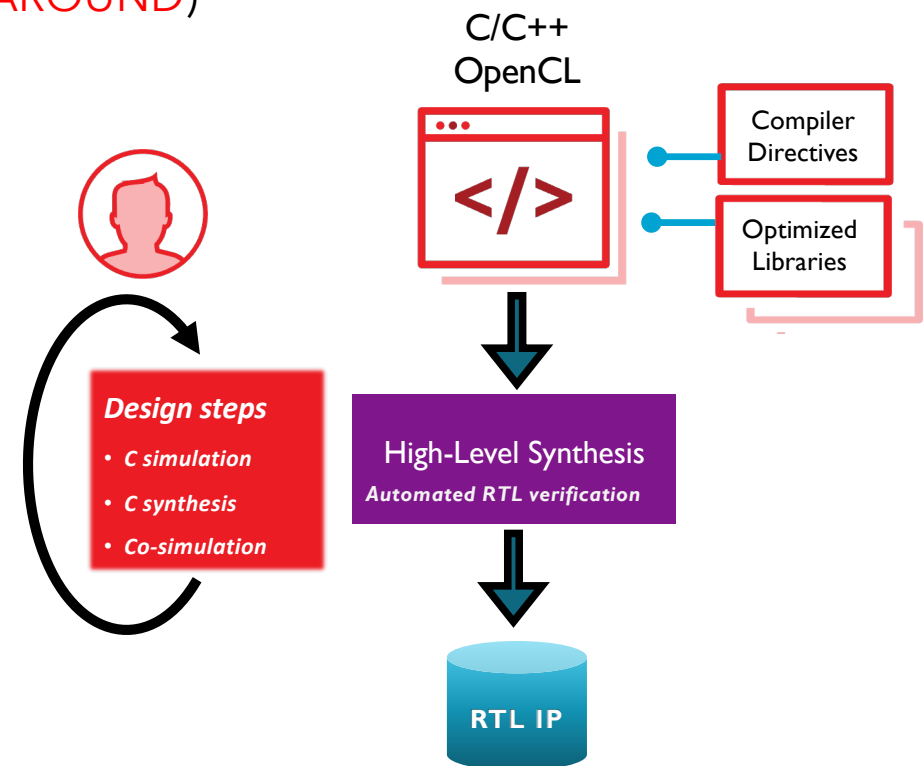
There are more C/C++/Python Developers than Verilog HDL Developers

HLS: Develop Custom-Accelerated Kernels



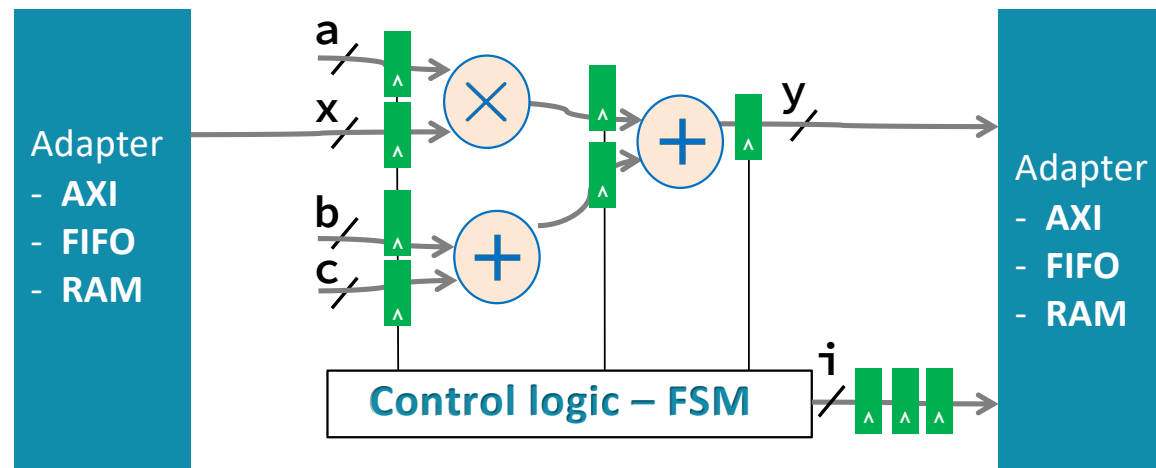
HLS: Design Accelerators in C / C++

- C/C++ Based Descriptions
- Higher Productivity (**LESS COSTS, FASTER TURNAROUND**)
 - Concise code
 - Optimized Libraries
 - Fast C simulation
 - Automated Simulation of Generated RTL



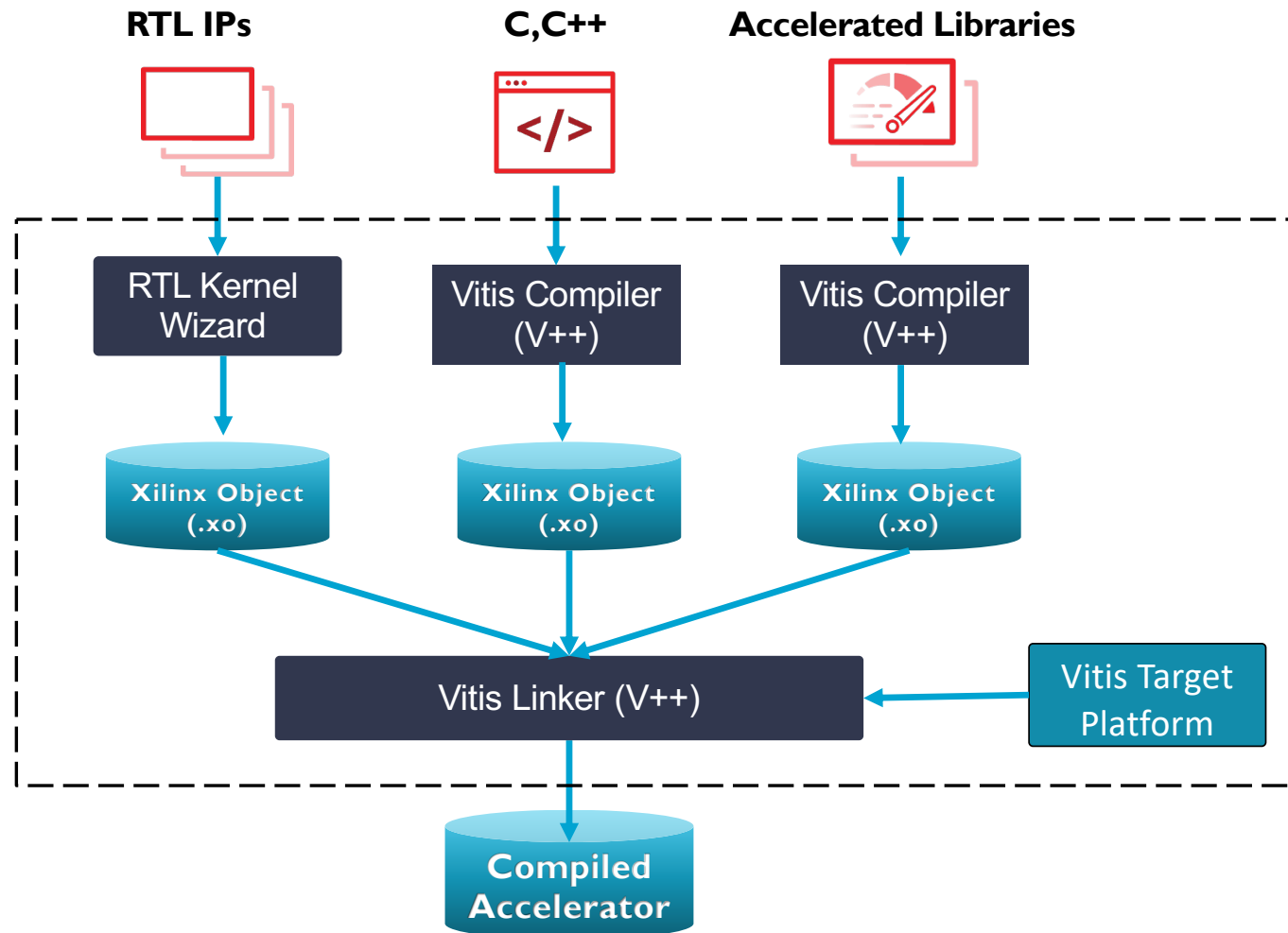
HLS: High-Level Program to H/W Circuits

```
int a, b, c, x, y;  
void f(int in[20], int out[20]) {  
    for (int i = 0; i < 20; i++) {  
        x = in[i]; y = a*x + b + c; out[i] = y;  
    }  
}
```



HLS Implements Interfaces and Control Logic

HLS: Compiling and Linking Accelerators



HLS: Running Accelerated Applications

No Actual Platforms or Accelerator Cards Required

Software Emulation	Hardware Emulation	Hardware Execution
Host application runs with a C/C++ or OpenCL model of the FPGA kernels	Host application runs with a simulated RTL model of the FPGA kernels	Host application runs with actual hardware implementation of the FPGA kernels
Verify Functional Correctness of System	Verify Host / Accelerator Integration, Get Performance Estimates	Validate System Runs Correctly with Desired Accelerated Performance
Fastest Turnaround Time, Quick Design Iterations	Best Debug Capabilities	Long Turnaround Time, Accelerated Performance

Open Computing Language (OpenCL) is a standard for parallel programming using CPUs, GPUs, FPGAs, DSPs, etc.

HLS: Use Open-Source Libraries



Domain-Specific Libraries



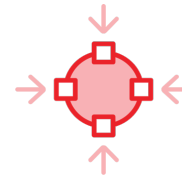
Vision & Image



Quantitative Finance



Data Analytics & Database



Data Compression

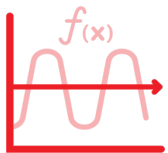


Data Security

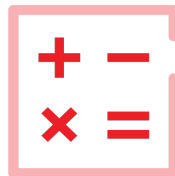


Partner Libraries

Common Libraries



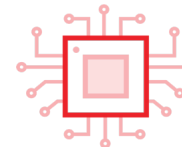
Math



Linear Algebra



Statistics



DSP



Data Management

500+ functions across multiple libraries for performance-optimized out-of-the-box acceleration

Today's Class Re-cap

- 4 FPGA Use Cases
- FPGA based SoC Architecture
- Zynq-7000 PS Customization
- Zynq-7000 PS Block Diagram + PS/PL Interfaces
- Zynq-7000 Boot Sequence
- High Level Synthesis (HLS)