Introduction

- Syllabus
- Industry Trend Towards Integration in Embedded System Design
- Introducing SoPC and Embedded System Design Flow
- Introducing EDA Tools and Lab Board

Embedded Systems Design in the Old Days

- Systems were built by using many discrete parts
  - FPGA, memory, and I/O
- Logic design tools from a variety of suppliers were (and still are) used to specify their individual functionalities as well as integrating them to function together.
- The CPU side of the product was (and still is) most often discrete part.

Embedded Systems Design in the Old Days

- The interfacing of the CPU to the logic, memory, and I/O systems is performed through the logic tools.
- The software side was (and still is) implemented by using specialized tools for code and OS development.
- The hardware and software interfacing work is performed by using a combination of these tools and, in some cases, co-verification or co-simulation tools.

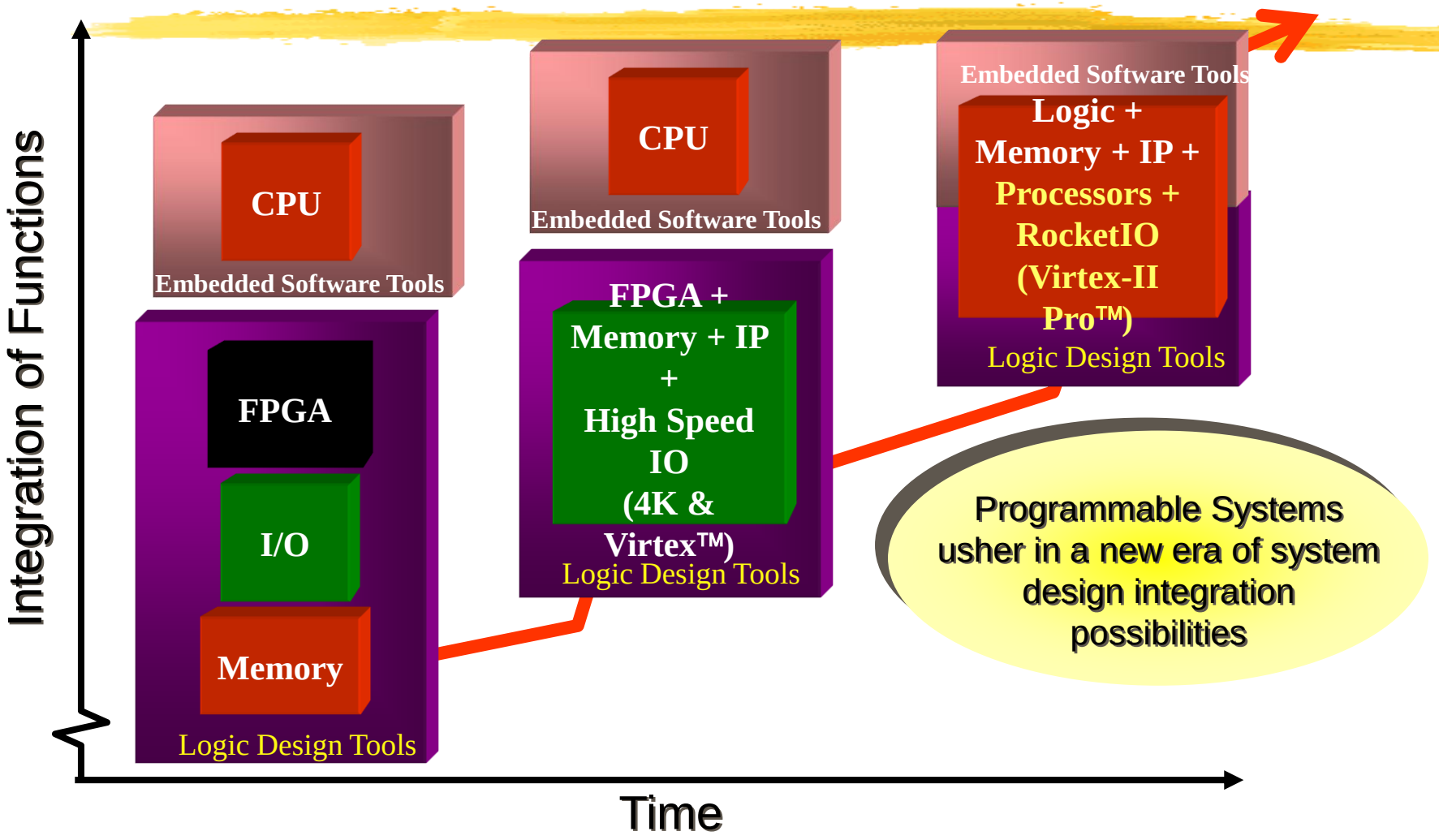
Changes that are Undergoing

- In the later part of the 1990s, many of the logic functions became integrated into single devices, such as Virtex™ families from Xilinx.
- Larger amounts of memory are still off-chip, but more memory became available as new families were introduced and densities increased.
- This enabled more of the logic design to be developed and implemented as a single unit rather than as discrete parts that required additional interfacing.

What's Happening Now

- With the advent of the Platform FPGA and SoC ASICs, merging much of the discrete functionality into a single part is now reality.
- Unlocking the full potential of fully hardware-and-software programmable systems: All Programmable SoC, both as an end product and as an advantageous development process, requires a coupling of the silicon, the hardware and software IP, and the development tools into a cohesive unit of parts and design solutions.

Integration in System Design



SoC Platform

System on Chip

System On Chip

Application

Specific

Integrated

Circuit

System On Programmable Chip

Field

Programmable

Gate

Array

We are here

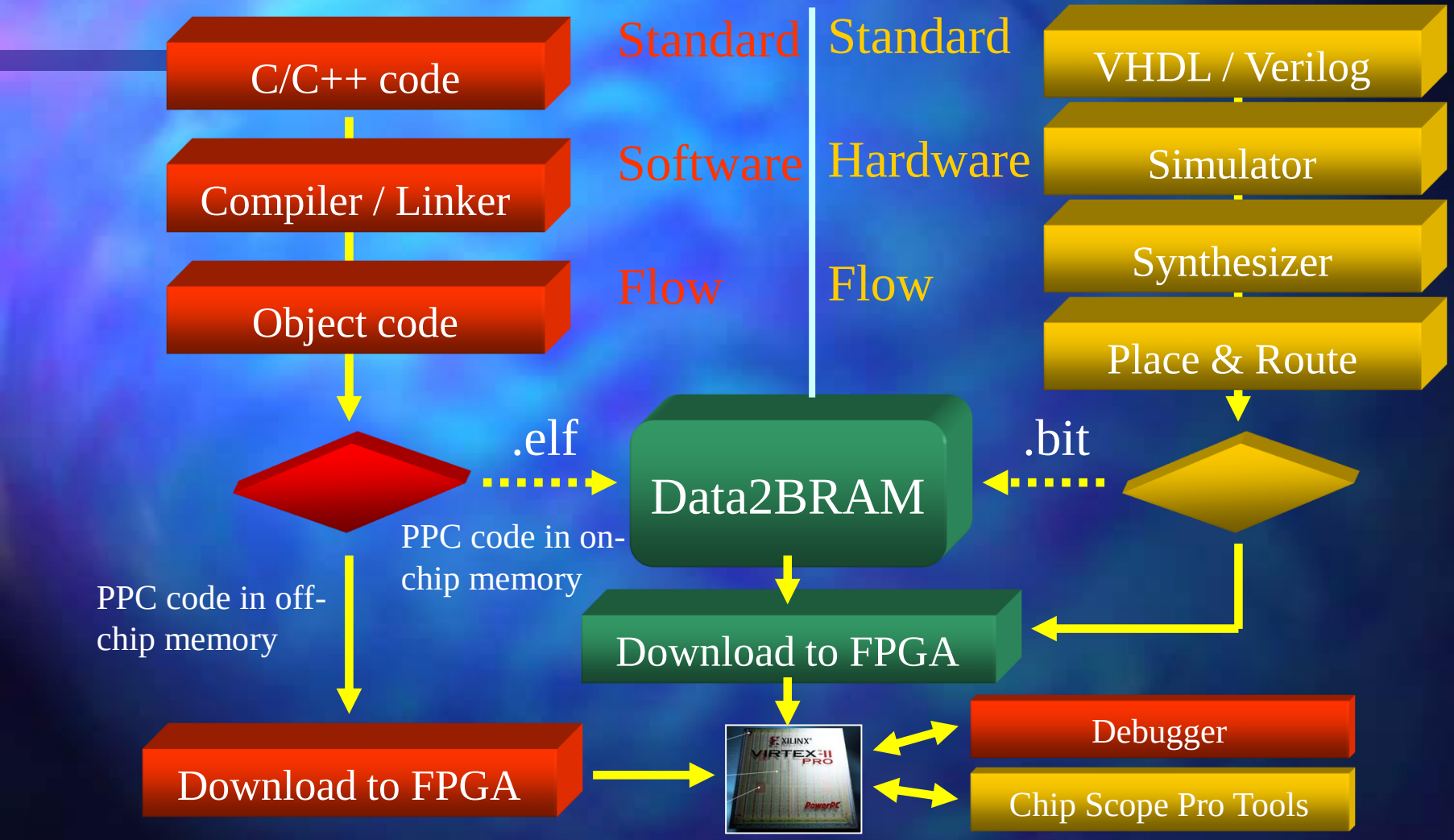
Hardware Design Process for SoPC

Choose chip with required hard cores inside

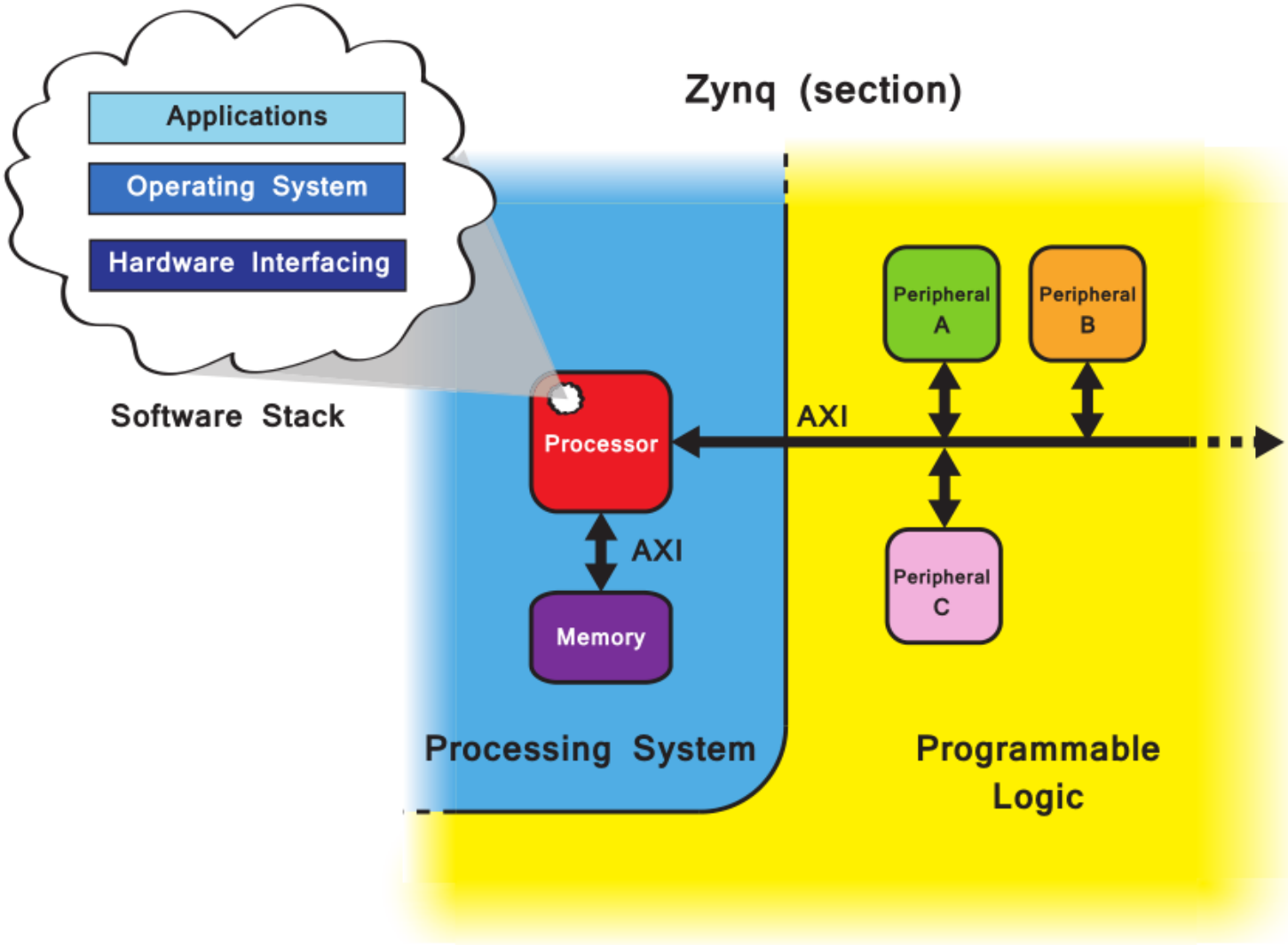
Add required soft cores

Add user logic

SoPC Design Tool Chain

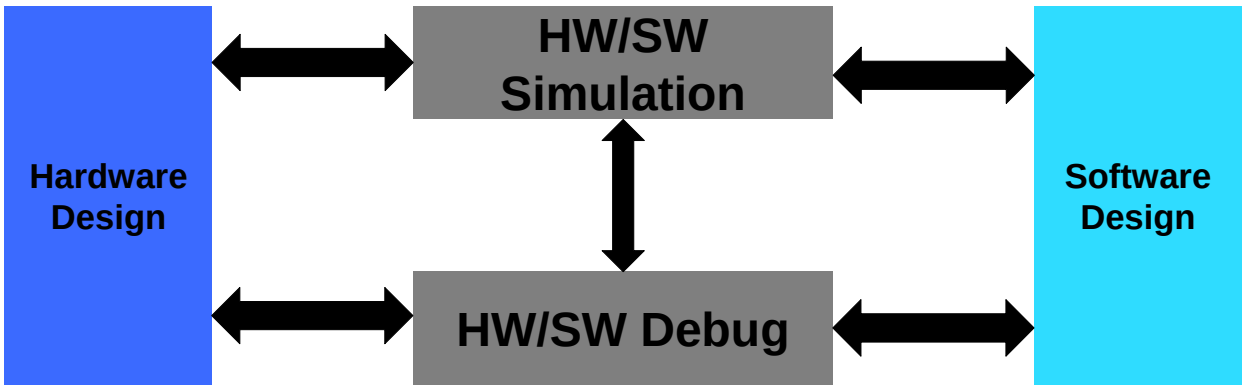


Vivado
– Xilinx New Generation of All Programmable Soc Design

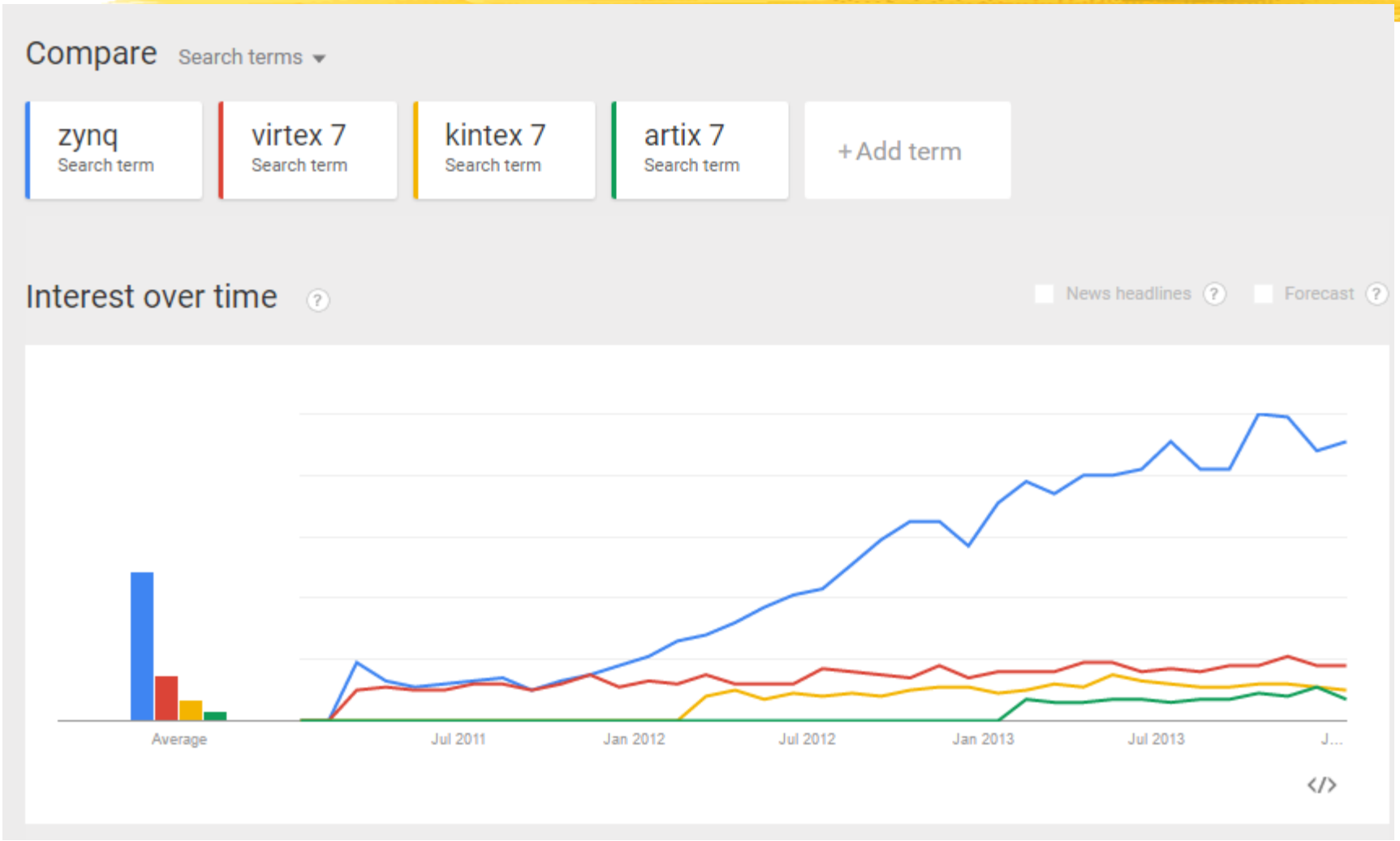


Vivado Components

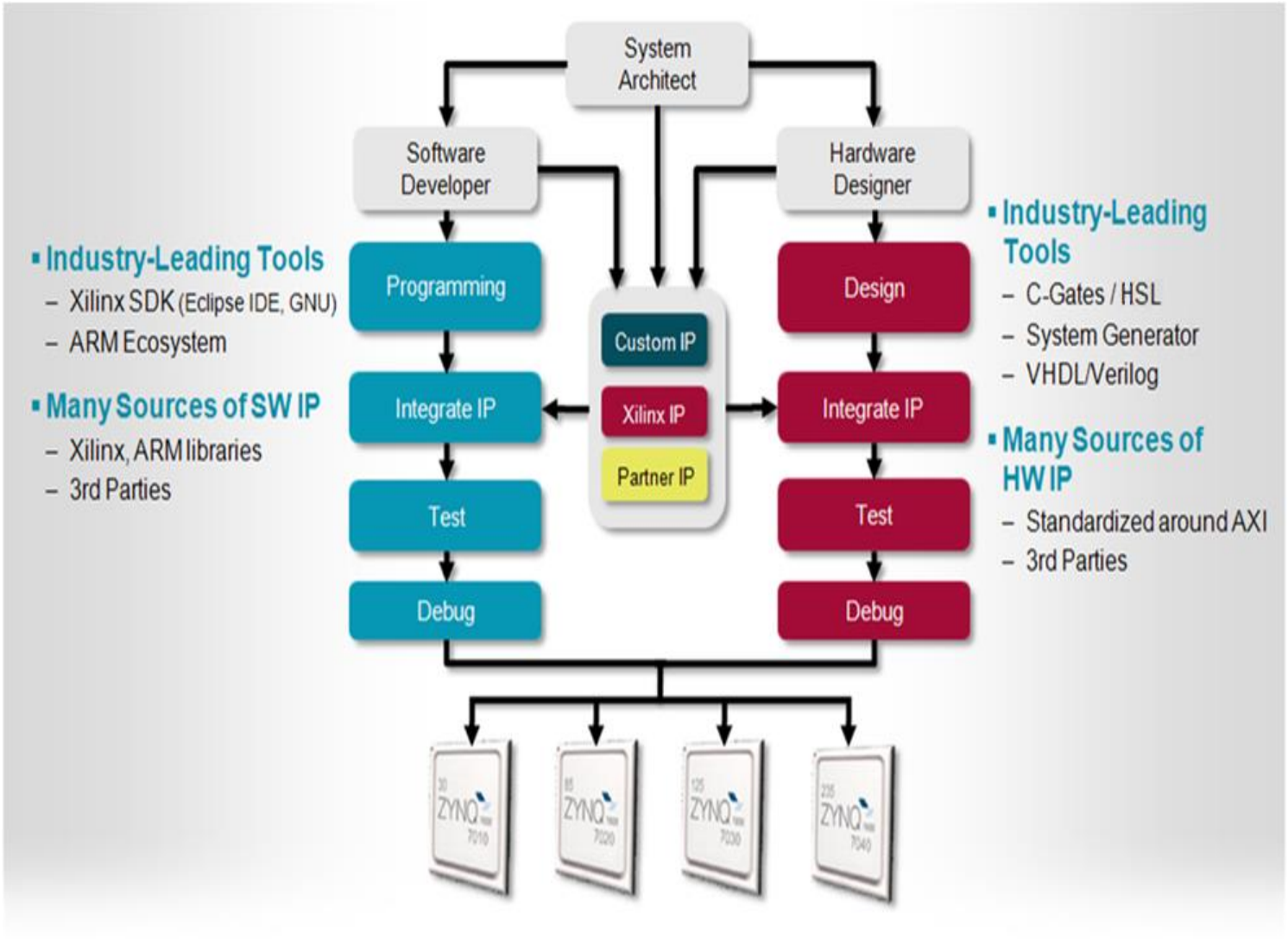
- **Vivado/IP Integrator**
  - Design environment for configuration of PS, and hardware design for PL
  - Hardware Platform (xml)
  - Platform, software, and peripheral simulation
  - Vivado logic analyzer integration
- **Software Development Kit (SDK)**
  - Project workspace
  - Hardware platform definition
  - Board Support Package (BSP)
  - Software application
  - Software debugging



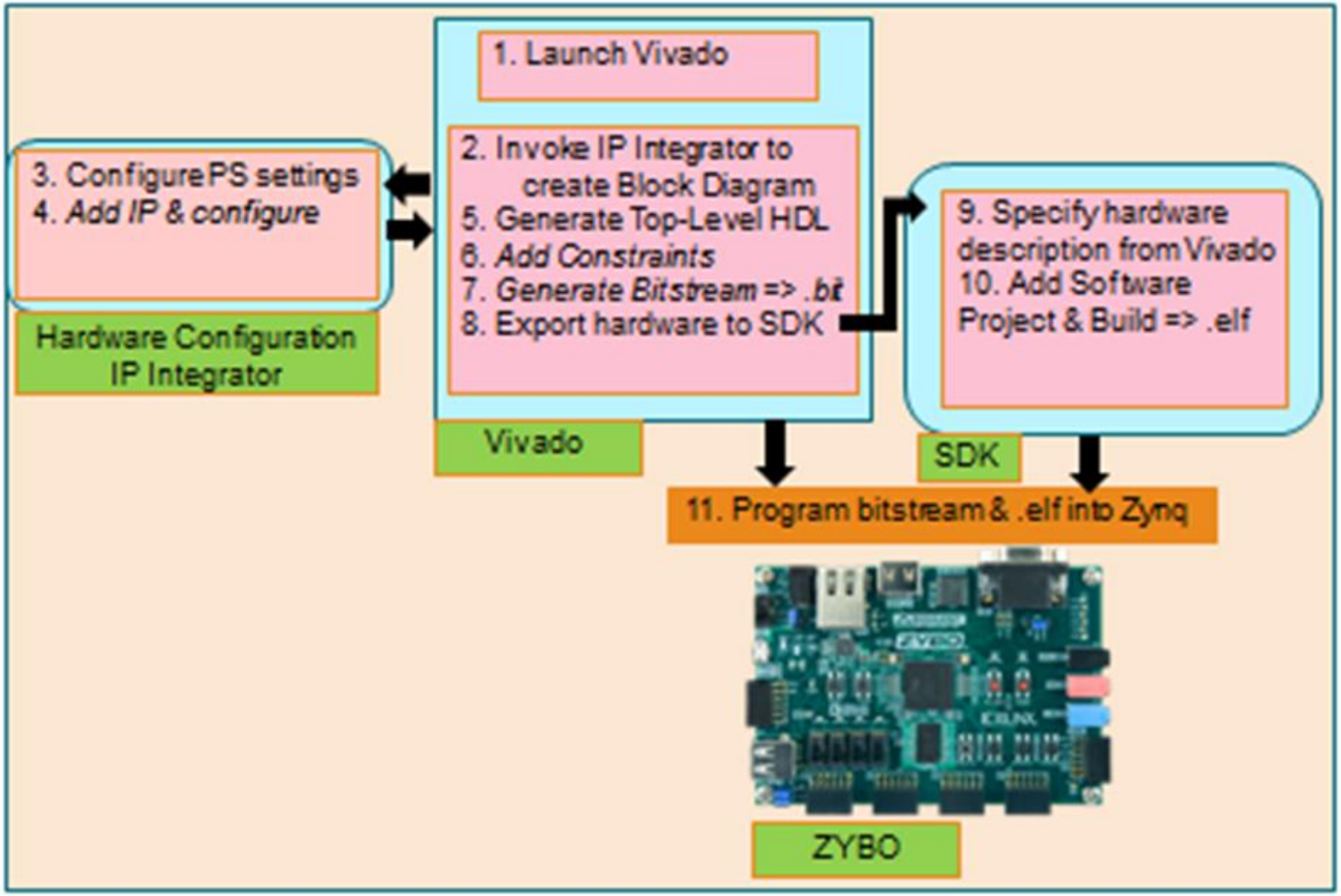
Zynq and the trend towards ARM-FPGA architectures



Embedded System Design Flow for Zynq-7000 AP SoC



Embedded System Design using Vivado



Course Objectives

- **After completing this course, you will be able to:**
  - Rapidly architect an embedded system targeting Zynq, and the AXI4 interface standard, using Vivado and IP Integrator
  - Extend the system by adding peripherals
    - Add Xilinx provided peripherals from the IP catalog
    - Create and add a custom peripheral using IP Integrator
  - Create and debug software applications
    - Create software applications in the Software Development Kit (SDK)
    - Debug an application on-chip using the GNU debugger via SDK

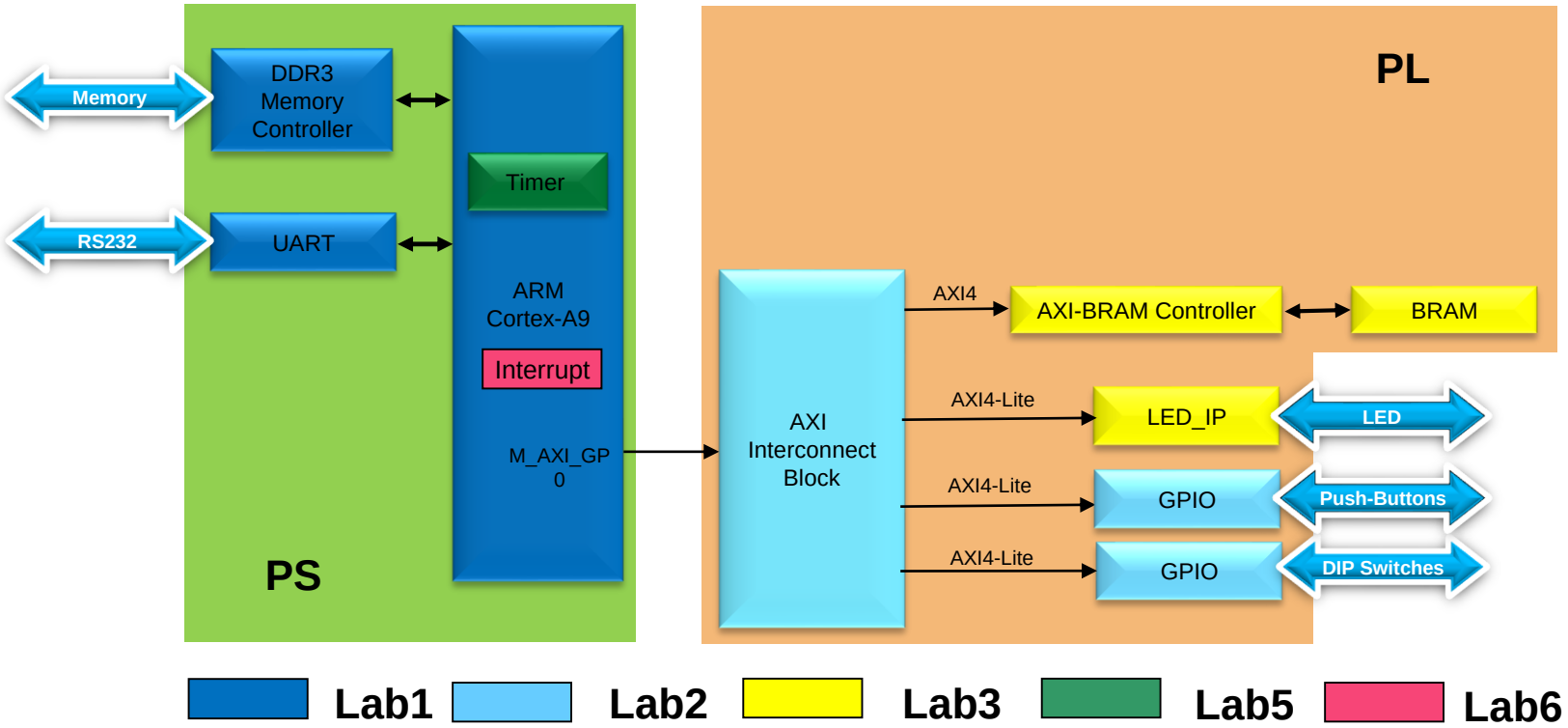
Topics to be Covered

- Embedded system design flow using Vivado 2015.2 System Edition
- Introducing FPGA and Zybo
- Introducing ARM Cortex-A9 processor architecture
- Basic embedded system designs, including GPIO, timers, interrupts, adding IPs and custom IPs, DMA, profiling and hardware accelerators, etc.
- Xilinx device driver architecture, adding custom IP into your embedded systems, writing device drivers for custom Ips.

Basic Labs

- **Lab 1**: Simple Hardware Design
- Zynq Architecture
- Extending the Embedded System into PL
- **Lab 2**: Adding IPs in Programmable Logic
- Adding Your Own Peripheral
- **Lab 3**: Creating and Adding Custom IP
- Software Development Environment
- **Lab 4**: Writing Basic Software Applications
- Software Development and Debugging
- **Lab 5**: Software Debugging Using SDK
- **Lab 6**: FPGA-Based Embedded System with Timer and Interrupt

ARM Cortex-A9 based Embedded System Design
Lab 1 through Lab 6



Lab4 uses hardware built in Lab3

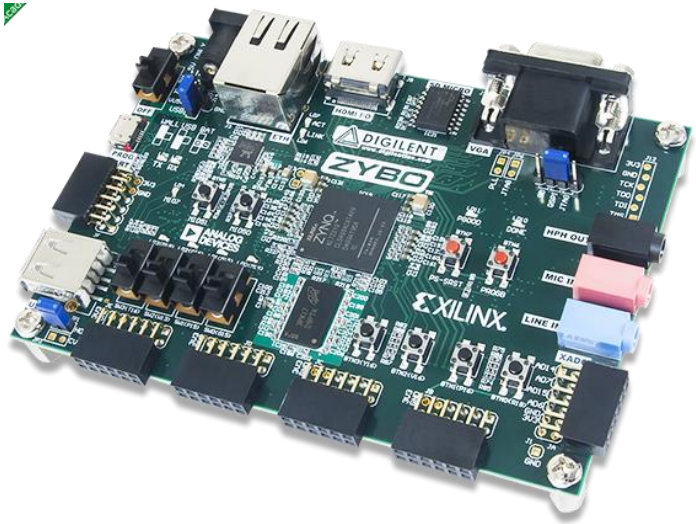
Prerequisites

- Basic C programming
- Basic understanding of processor-based system
- Basic HDL knowledge

Platform Support

- **Vivado System Edition 2015.2**
- **Xilinx University board**
  - Zybo
- **Supported Operating Systems**
  - Windows 7 SP1 Professional (64 Bit)
  - Windows 8.1 Professional (64 Bit)
  - Red Hat Enterprise Linux 6.5 – 6.6 (64 Bit)
  - Red Hat Enterprise Linux 7.0 (64 Bit)
  - SUSE Linux Enterprise 12.0 (64 Bit)
  - Centos Linux 7.0 (64 Bit)
  - Ubuntu Linux 14.04 LTS (64 Bit)

Lab Board - Zybo Zynq-7000 ARM/FPGA SoC Trainer Board



- Xilinx Zynq-7000: 28,000 logic cells
- 240 KB Block RAM
- On-chip dual channel, 12-bit, 1 MSPS analog-to-digital converter (XADC)
- 650 MHz dual-core Cortex™-A9 processor
- On-board JTAG programming and UART to USB converter
- High-bandwidth peripheral controllers: 1G Ethernet, USB 2.0, SDIO
- Low-bandwidth peripheral controller: SPI, UART, I2C
- Dual-role (Source/Sink) HDMI port 16-bits per pixel VGA output port
- Trimode (1Gbit/100Mbit/10Mbit) Ethernet PHY
- OTG USB 2.0 PHY (supports host and device)
- External EEPROM (programmed with 48-bit globally unique EUI-48/64™ compatible identifier)
- Audio codec with headphone out, microphone and line in jacks
- GPIO: 6 pushbuttons, 4 slide switches, 5 LEDs