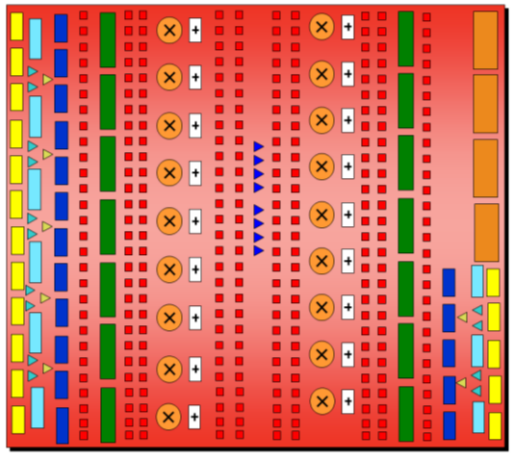
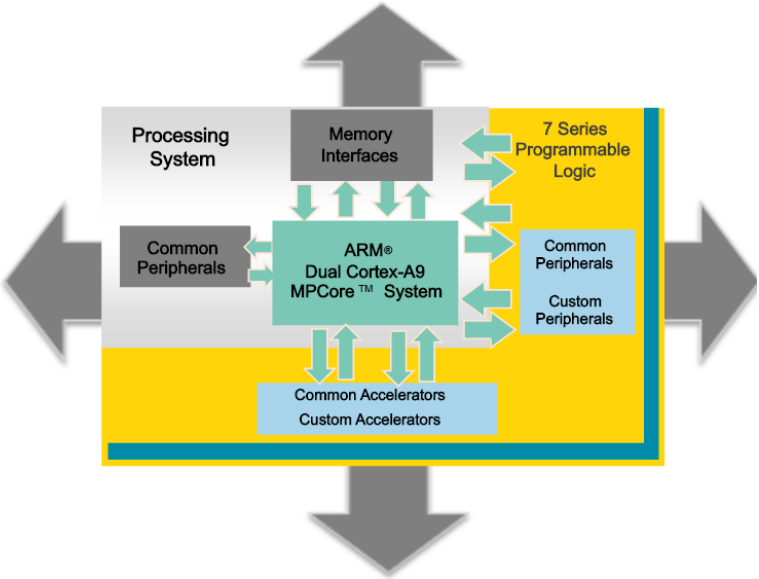
DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

A2A
Algorithms to Architecture

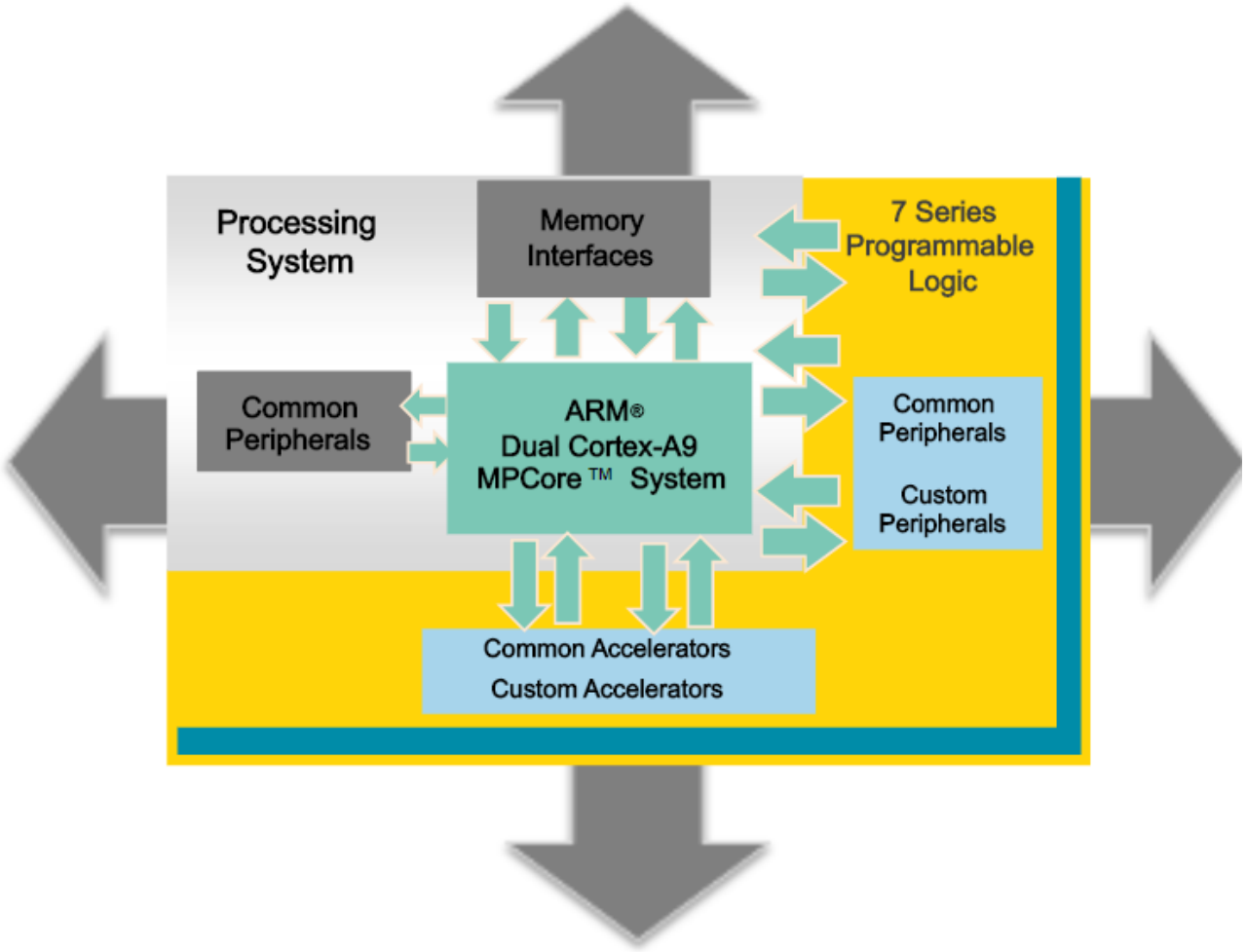
ECE 270: Embedded Logic Design

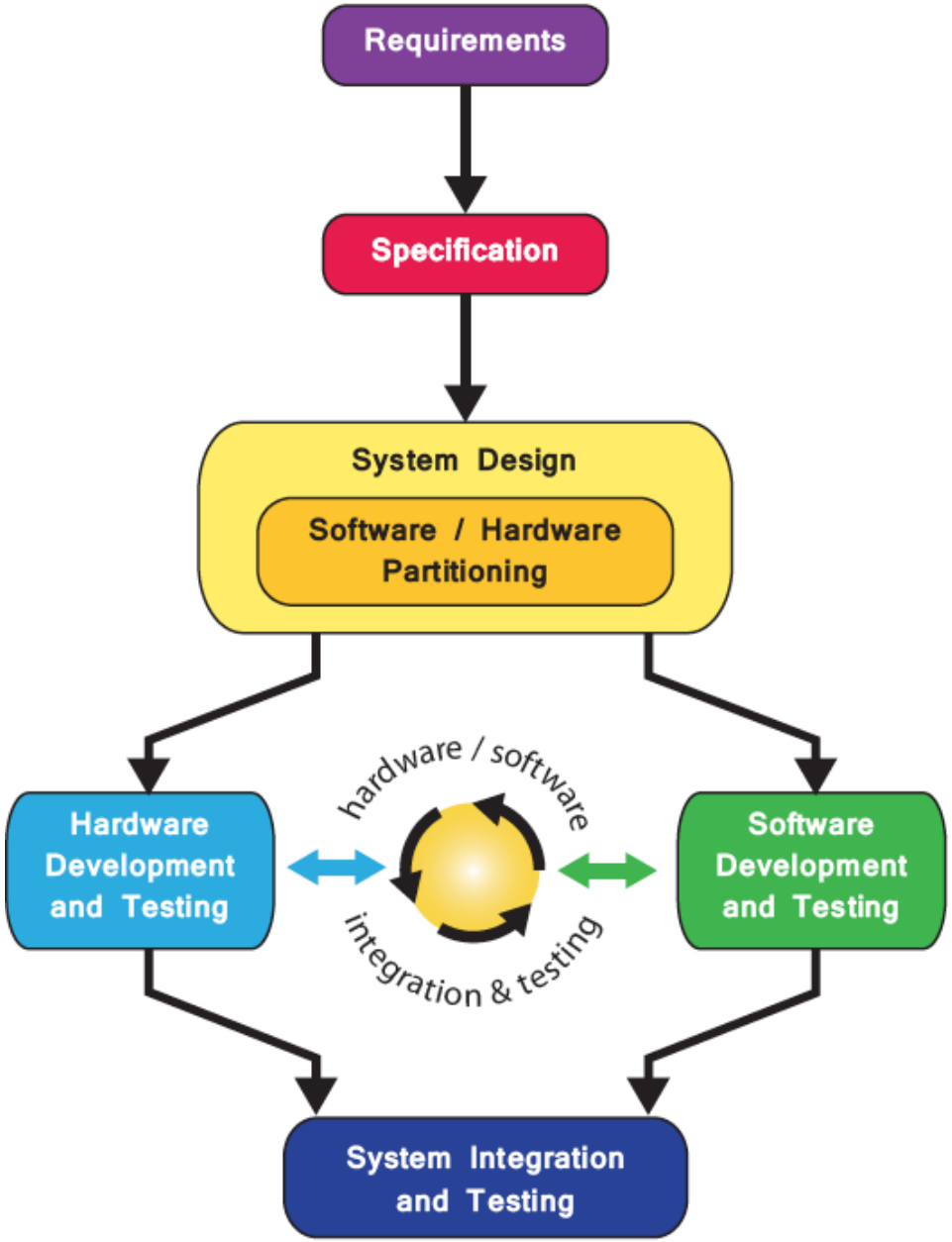


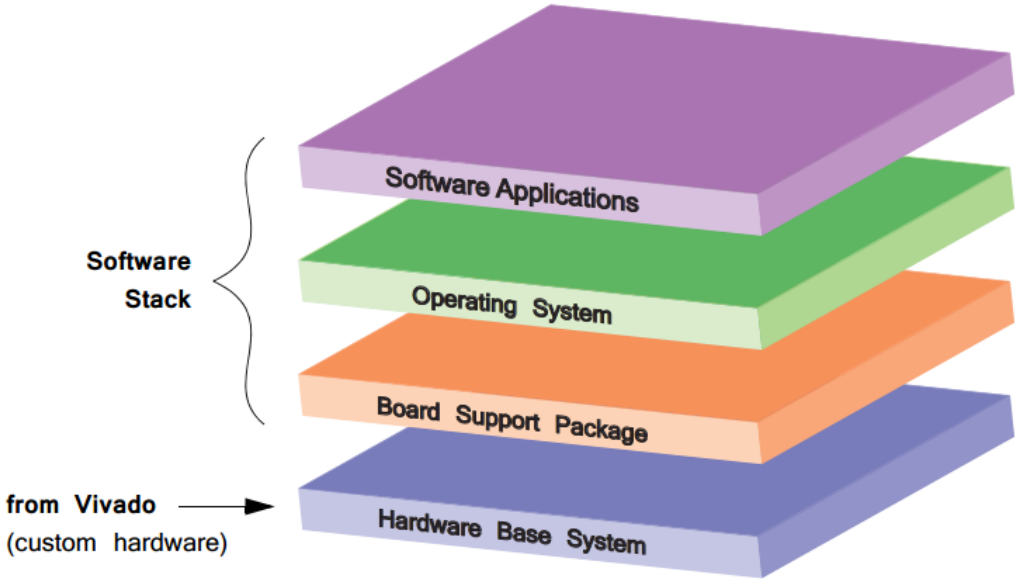


Zynq Boot Process

Zynq Design Flow

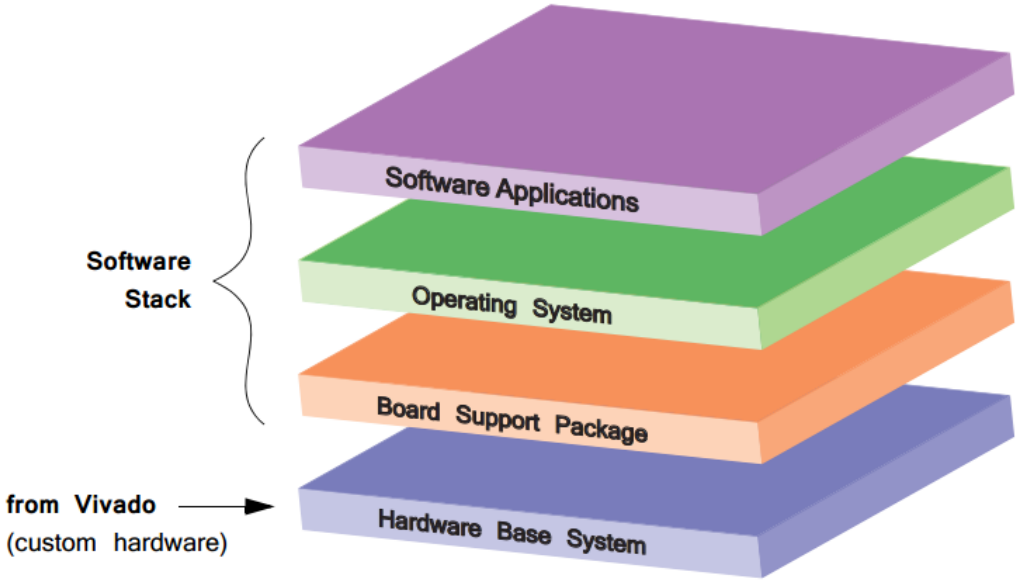






- ***Board Support Package (BSP)*:** Set of low-level drivers and functions that are used by the next layer up (the *Operating System or baremetal*) to communicate with the hardware
- **Software Applications** run on top of the Operating System — these collectively represent the uppermost layer in the software stack

- SDK provides the environment for creating BSPs, and developing and testing software for deployment in the upper layers.
- BSP (includes hardware parameters, device drivers, and low-level OS functions) should be refreshed if changes are made to the hardware base system.
- The purpose of ELF files is to program the PS, while BIT files are used to program the PL.
- Not all designs require both an Executable Linkable Format (ELF, *.elf) file and a BIT (*.bit) file to configure the device.
- If only one part of the Zynq device is used (PS or PL), then only the corresponding file type is needed.



- ***Board Support Package (BSP)*:** Set of low-level drivers and functions that are used by the next layer up (the *Operating System or baremetal*) to communicate with the hardware
- **Software Applications** run on top of the Operating System — these collectively represent the uppermost layer in the software stack

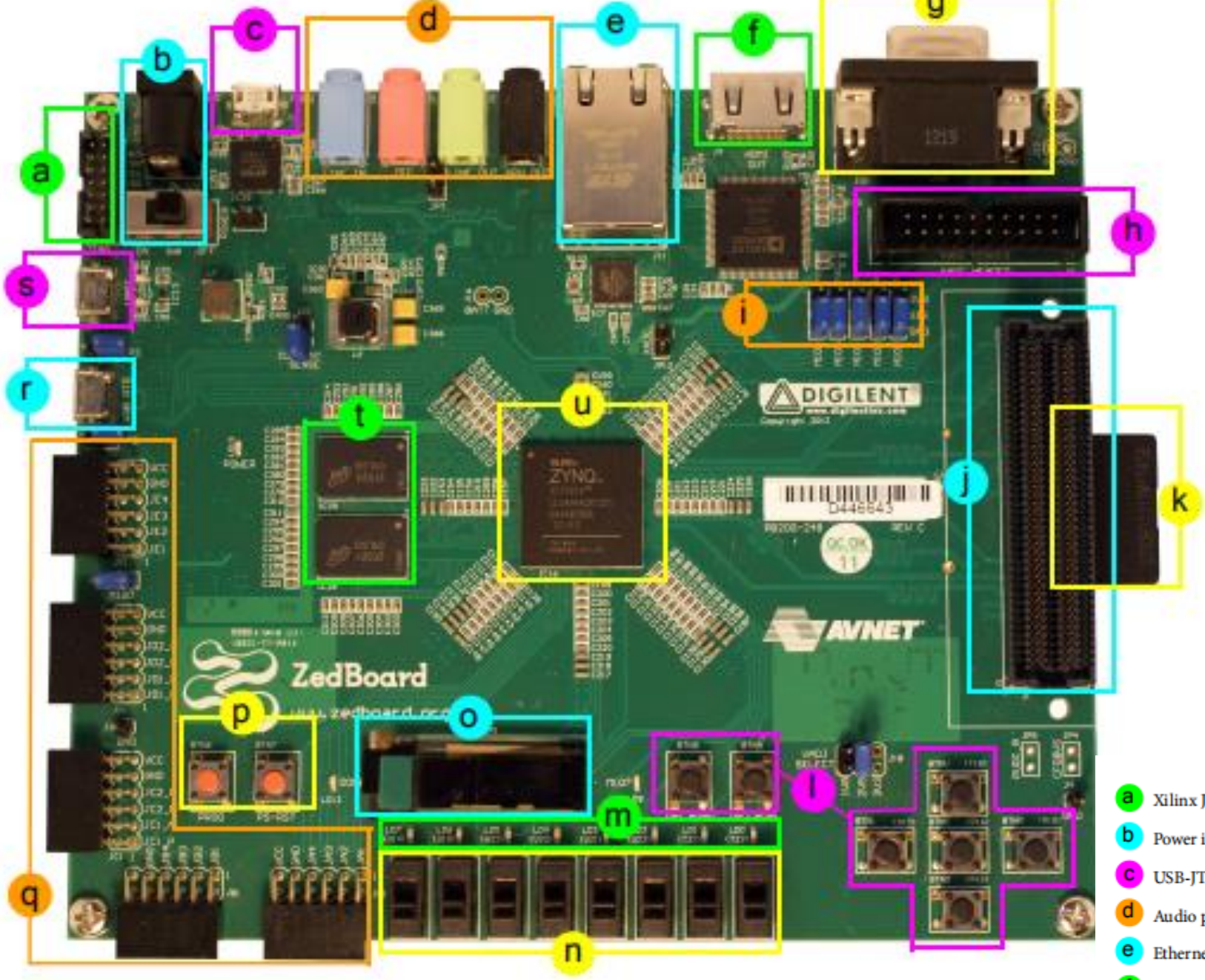
- When creating the software stack, the first design choice is usually to decide on the OS to deploy: 1) Fully-fledged OS such as Linux or Android, 2) Embedded OS, 3) Real-Time Operating System (RTOS) for deterministic, time-critical applications, 4) Standalone, a 'light' OS including only the basic functions.
- It is also possible for applications to communicate directly with hardware, and this is often referred to as a '**baremetal**' application.
- With two processor cores available, there is also the possibility of deploying two different types of OS, one on each core.

Zynq Boot and Configuration

- The processors in the PS always boot first, allowing a software centric approach for PL configuration.
- The PL can be configured as part of the boot process or configured at some point in the future.
- Additionally, the PL can be completely reconfigured or used with dynamic partial reconfiguration (DPR).

Zynq Boot and Configuration

- The boot process is multi-stage and minimally includes the boot ROM and the first-stage boot loader (FSBL).
- For example, Zynq-7000 AP SoC includes a factory-programmed boot ROM (stage 0 boot code) that is not user accessible.
- The main tasks of the BootROM (initialized at power-on) are to configure one of the ARM core, read the mode pins to determine the primary boot device, copy the Boot Image FSBL from the boot device to the on-chip memory (OCM), and then branch the code execution to the OCM. It finishes once it is satisfied that it can execute the FSBL.
- PL is not configured by BootROM.
- FSBL is typically stored in one of the flash memory or can be download through the JTAG

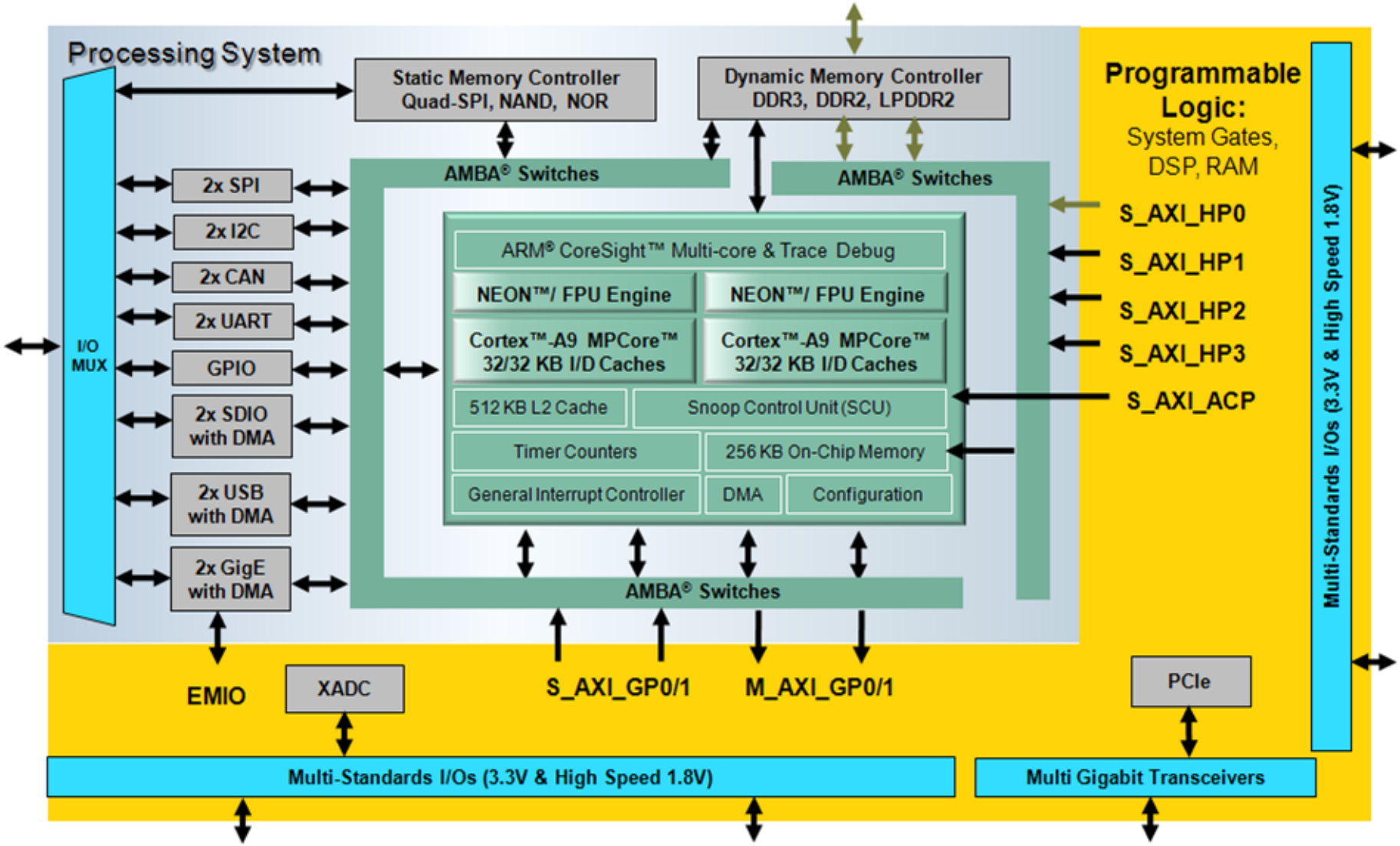


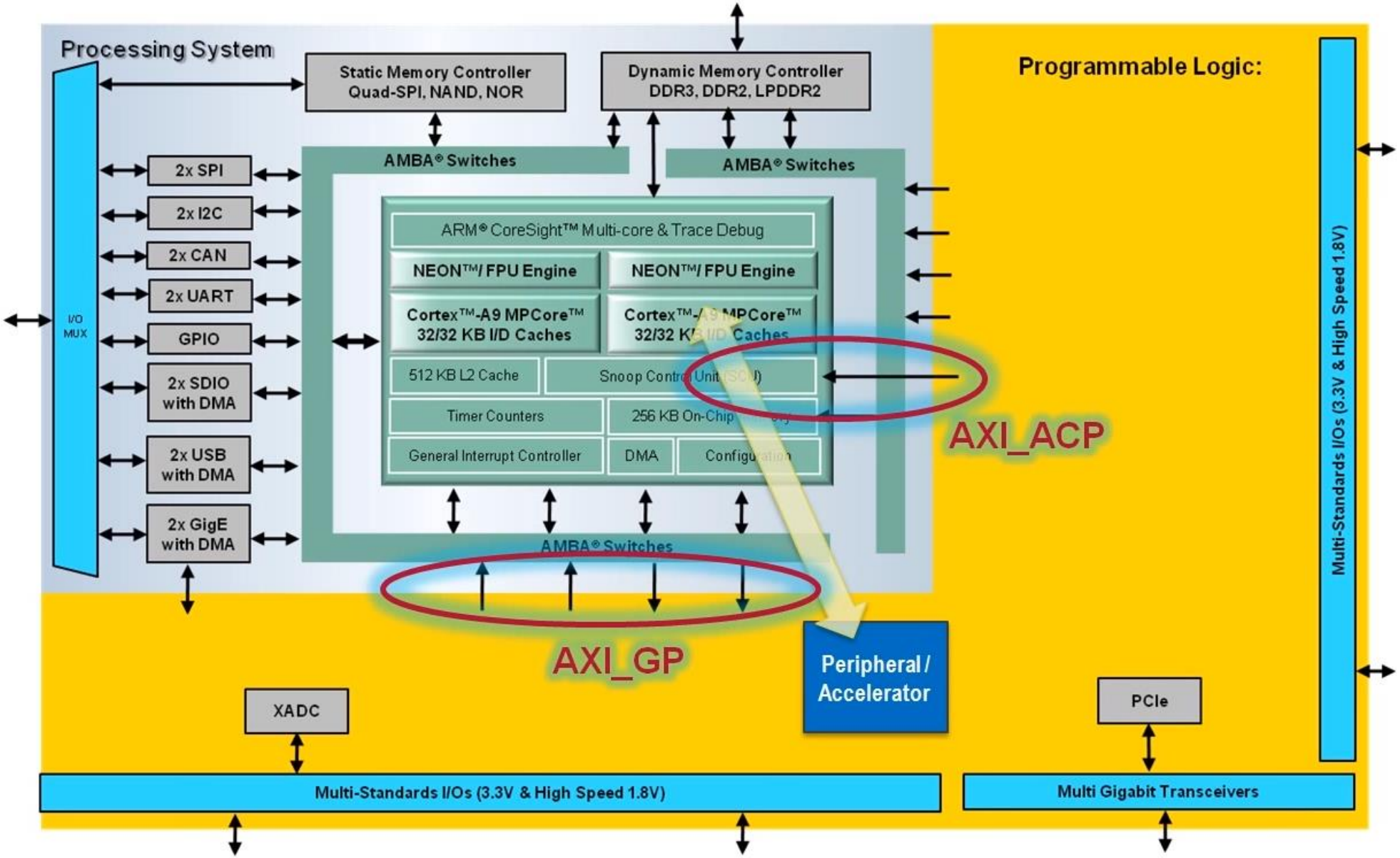
a Xilinx JTAG connector
b Power input and switch
c USB-JTAG (programming)
d Audio ports
e Ethernet port
f HDMI port (output)
g VGA port
h XADC header port
i Configuration jumpers
j FMC connector
k SD card (underside)
l User push buttons
m LEDs
n Switches
o OLED display
p Prog & reset push buttons
q 5 x Pmod connector ports
r USB-OTG peripheral port
s USB-UART port
t DDR3 memory
u Zynq device (+ heatsink)

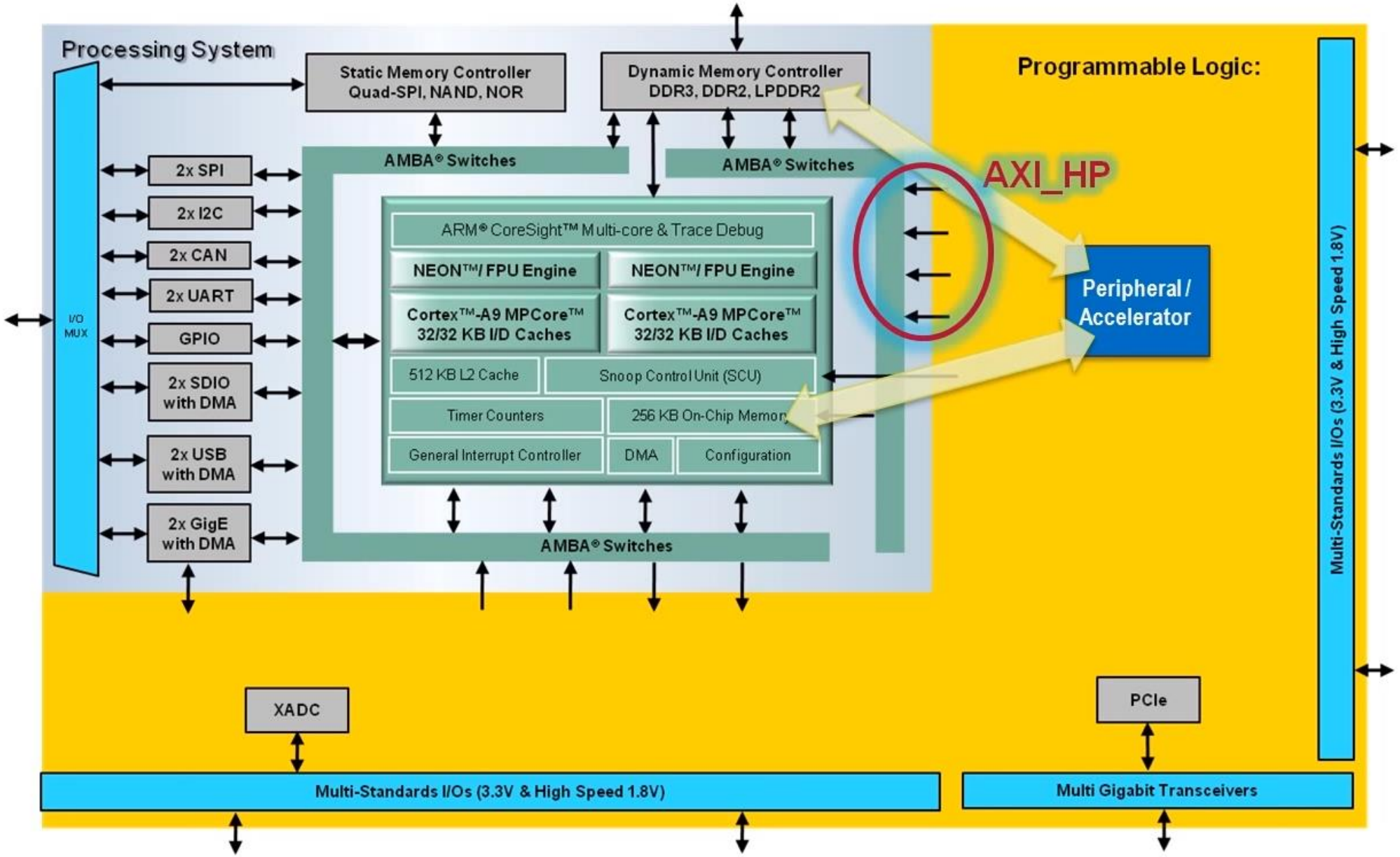
Zynq Boot and Configuration

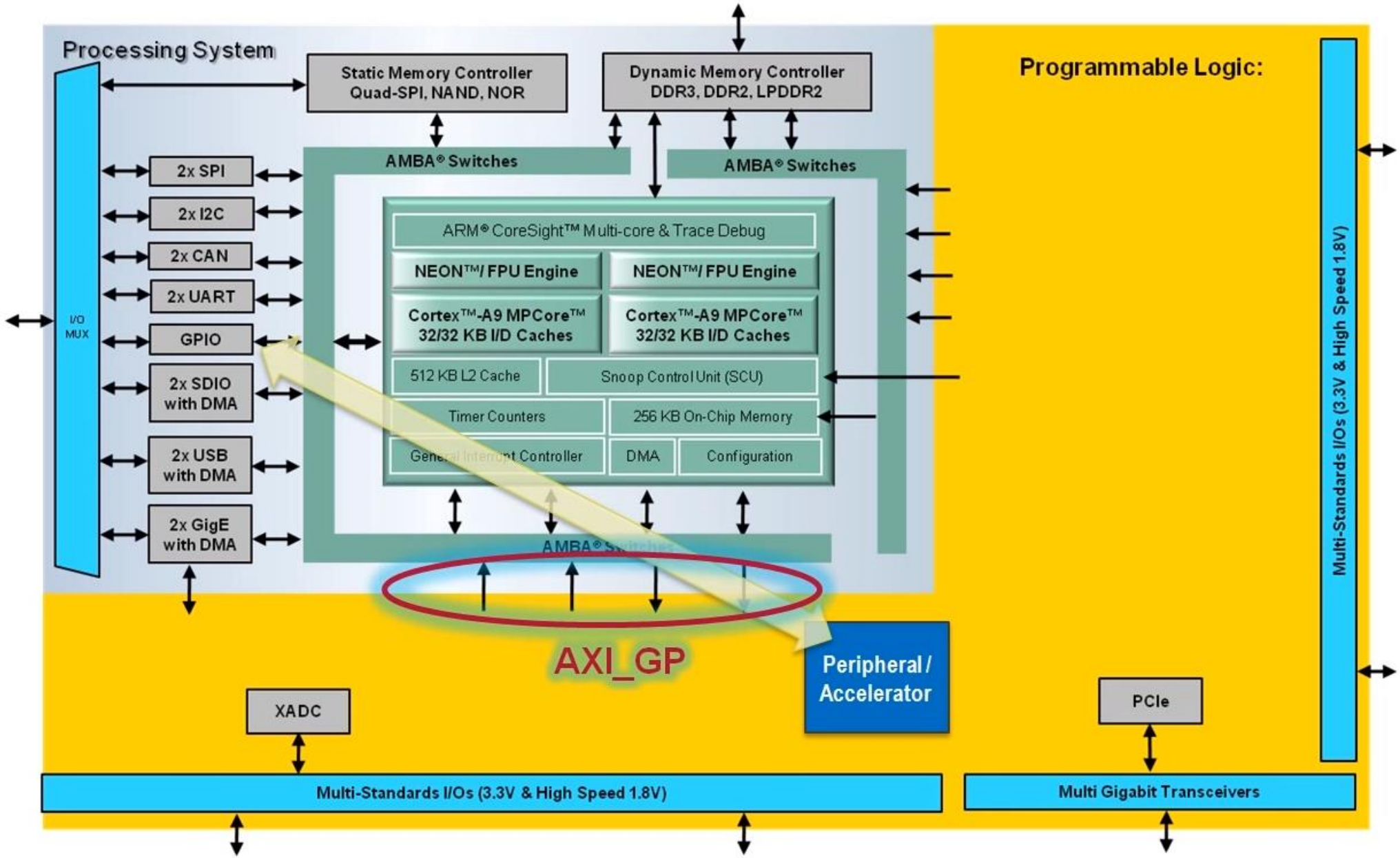
- The FSBL/User code executes after the BootROM is finished.
- The FSBL/User code reconfigures the PS as needed and optionally configures the PL.
- The FSBL/User code operations:
  - Initialize the PS using the PS7 Init data that is generated by Vivado tools (MIO, DDR, etc.)
  - Program the PL using a bitstream (if provided).
  - Load the second stage bootloader or bare-metal application code into DDR memory.
  - Hand off system control to the second stage bootloader or bare-metal application.

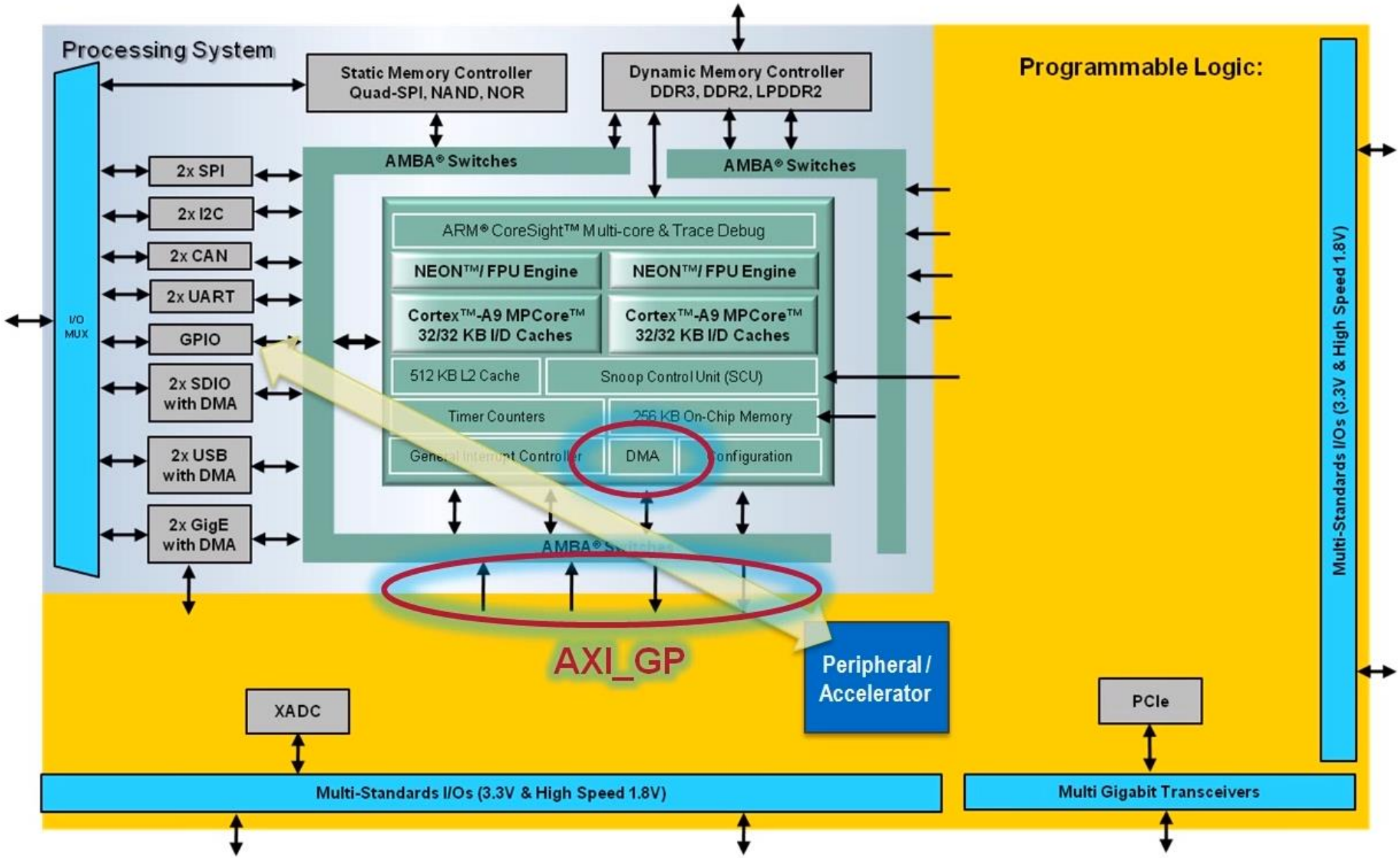
Zynq Architecture: PS and PL



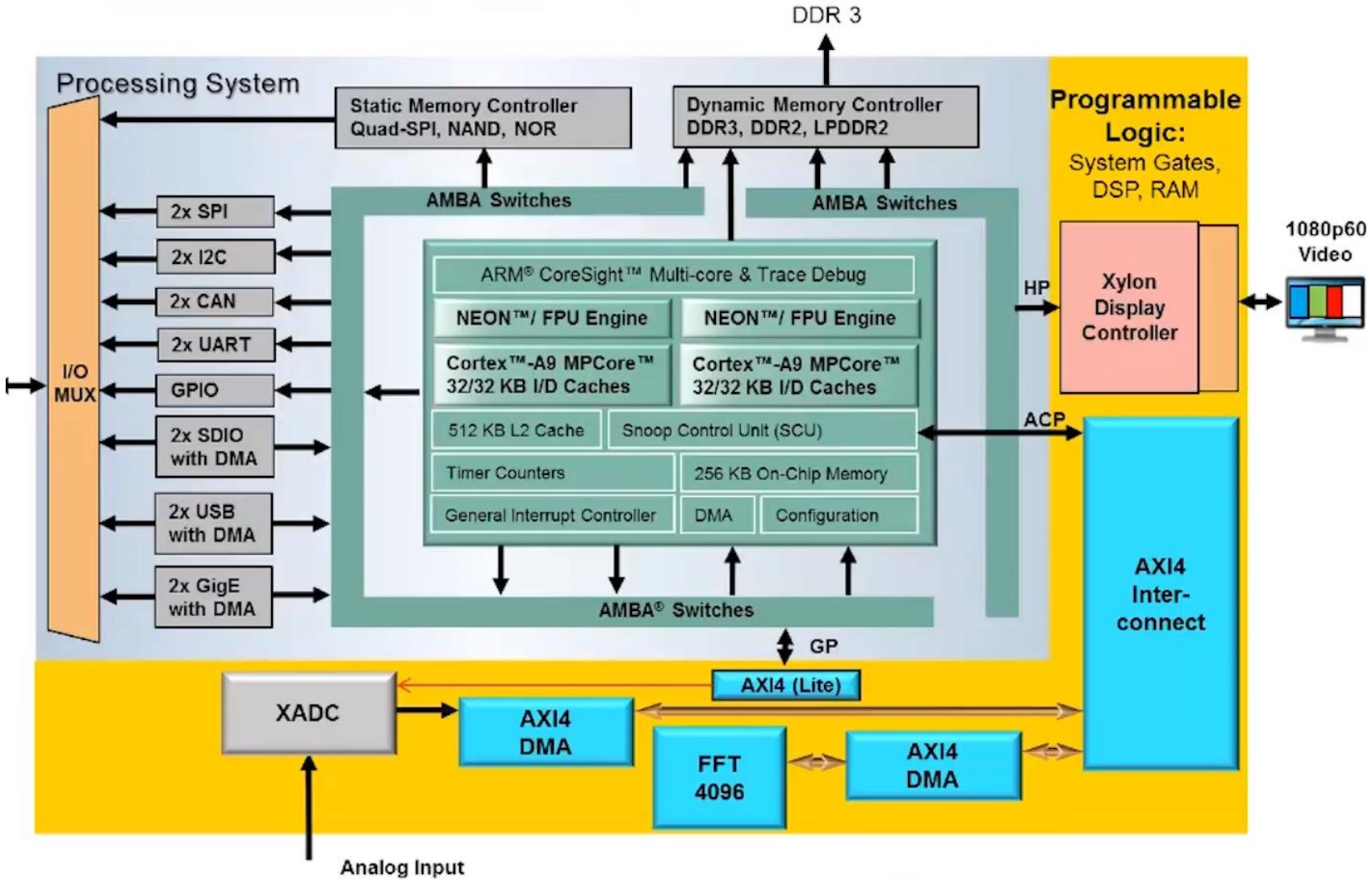

Processing System
Static Memory Controller
Quad-SPI, NAND, NOR
Dynamic Memory Controller
DDR3, DDR2, LPDDR2
Programmable Logic:
I/O MUX
2x SPI
2x I2C
2x CAN
2x UART
GPIO
2x SDIO with DMA
2x USB with DMA
2x GigE with DMA
AMBA® Switches
AMBA® Switches
AMBA® Switches
ARM® CoreSight™ Multi-core & Trace Debug
NEON™/ FPU Engine
NEON™/ FPU Engine
Cortex™-A9 MPCore™ 32/32 KB I/D Caches
512 KB L2 Cache
Timer Counters
General Interrupt Controller
DMA
AXI_ACP
AXI_GP
Peripheral / Accelerator
XADC
PCIe
Multi-Standards I/Os (3.3V & High Speed 1.8V)
Multi Gigabit Transceivers
Multi-Standards I/Os (3.3V & High Speed 1.8V)


Processing System
Static Memory Controller
Quad-SPI, NAND, NOR
Dynamic Memory Controller
DDR3, DDR2, LPDDR2
Programmable Logic:
AMBA® Switches
AMBA® Switches
AXI_HP
2x SPI
2x I2C
2x CAN
2x UART
GPIO
2x SDIO
with DMA
2x USB
with DMA
2x GigE
with DMA
I/O
MUX
ARM® CoreSight™ Multi-core & Trace Debug
NEON™/ FPU Engine
NEON™/ FPU Engine
Cortex™-A9 MPCore™
32/32 KB I/D Caches
Cortex™-A9 MPCore™
32/32 KB I/D Caches
512 KB L2 Cache
Snoop Control Unit (SCU)
Timer Counters
256 KB On-Chip Memory
General Interrupt Controller
DMA
Configuration
AMBA® Switches
Peripheral /
Accelerator
Multi-Standards I/Os (3.3V & High Speed 1.8V)
XADC
PCIe
Multi-Standards I/Os (3.3V & High Speed 1.8V)
Multi Gigabit Transceivers

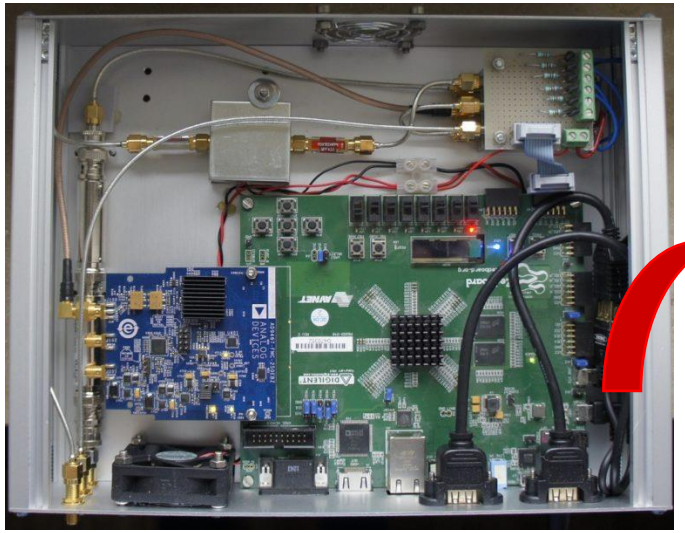

Processing System
Static Memory Controller
Quad-SPI, NAND, NOR
Dynamic Memory Controller
DDR3, DDR2, LPDDR2
Programmable Logic:
AMBA® Switches
AMBA® Switches
2x SPI
2x I2C
2x CAN
2x UART
GPIO
2x SDIO with DMA
2x USB with DMA
2x GigE with DMA
I/O MUX
ARM® CoreSight™ Multi-core & Trace Debug
NEON™/ FPU Engine
NEON™/ FPU Engine
Cortex™-A9 MPCore™ 32/32 KB I/D Caches
Cortex™-A9 MPCore™ 32/32 KB I/D Caches
512 KB L2 Cache
Snoop Control Unit (SCU)
Timer Counters
256 KB On-Chip Memory
DMA
Configuration
AXI_GP
Peripheral / Accelerator
XADC
PCIe
Multi-Standards I/Os (3.3V & High Speed 1.8V)
Multi Gigabit Transceivers
Multi-Standards I/Os (3.3V & High Speed 1.8V)

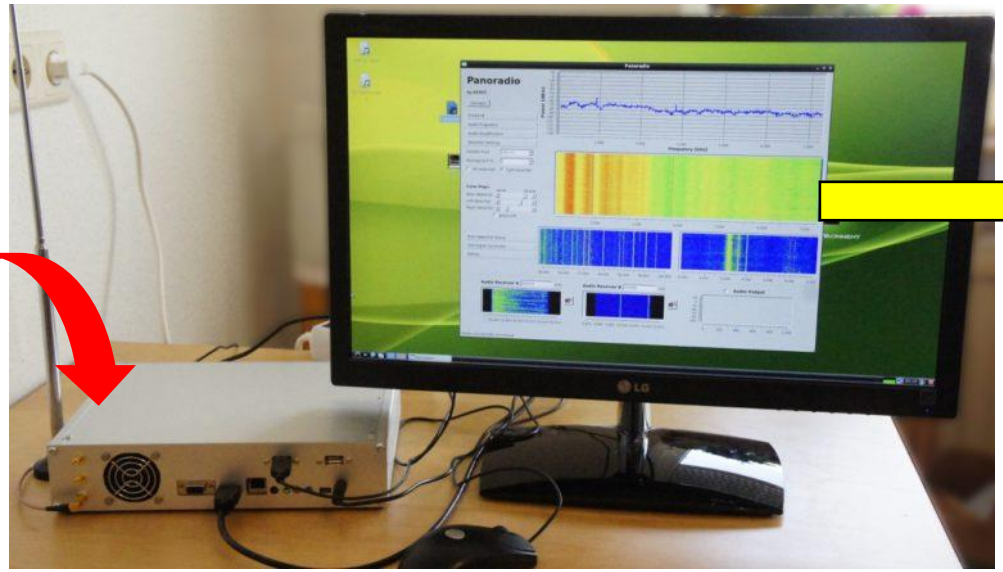

Processing System
Static Memory Controller
Quad-SPI, NAND, NOR
Dynamic Memory Controller
DDR3, DDR2, LPDDR2
Programmable Logic:
AMBA® Switches
AMBA® Switches
2x SPI
2x I2C
2x CAN
2x UART
GPIO
2x SDIO
with DMA
2x USB
with DMA
2x GigE
with DMA
I/O
MUX
ARM® CoreSight™ Multi-core & Trace Debug
NEON™/ FPU Engine
NEON™/ FPU Engine
Cortex™-A9 MPCore™
32/32 KB I/D Caches
Cortex™-A9 MPCore™
32/32 KB I/D Caches
512 KB L2 Cache
Snoop Control Unit (SCU)
Timer Counters
256 KB On-Chip Memory
DMA
AMBA®
AXI_GP
Peripheral /
Accelerator
XADC
PCIe
Multi-Standards I/Os (3.3V & High Speed 1.8V)
Multi Gigabit Transceivers
Multi-Standards I/Os (3.3V & High Speed 1.8V)

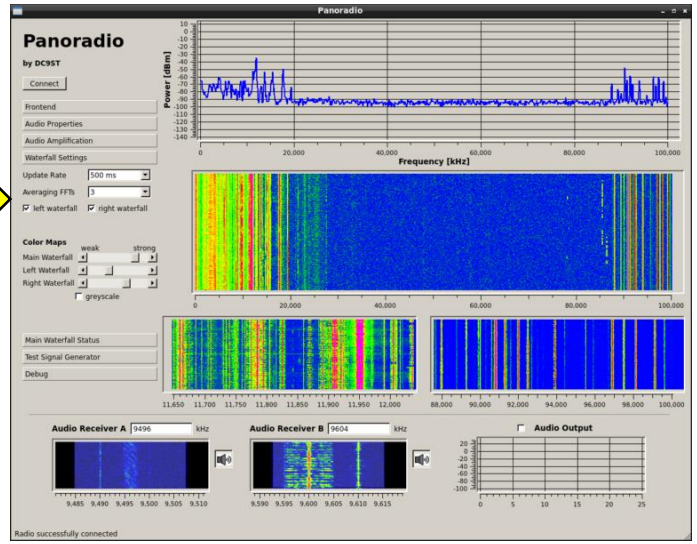
Spectrum Analyzer



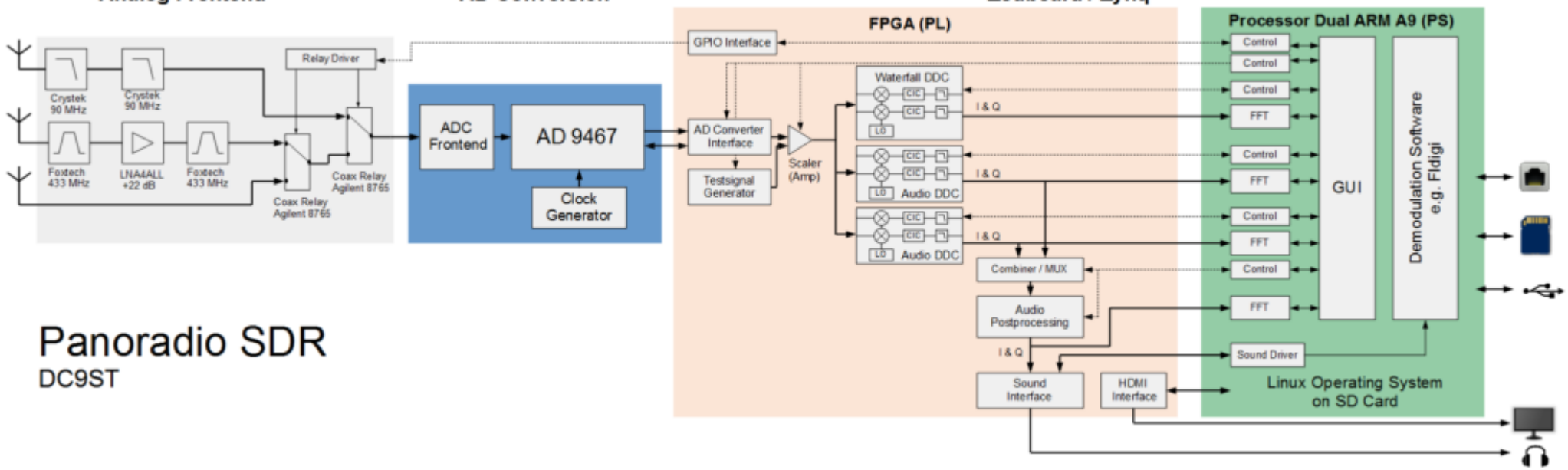
Software Defined Radio







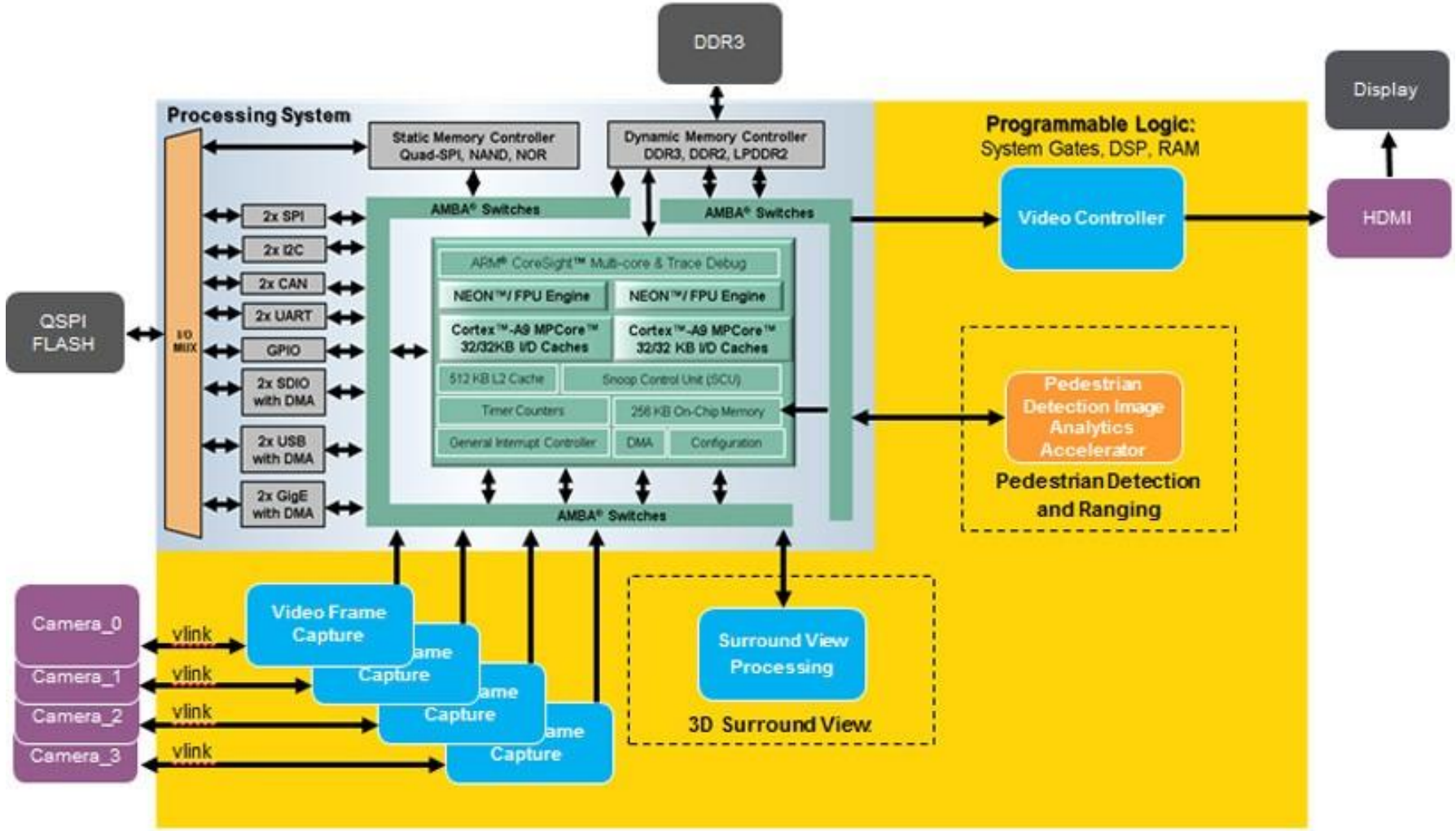
Analog Frontend **AD Conversion** **Zedboard / Zynq**



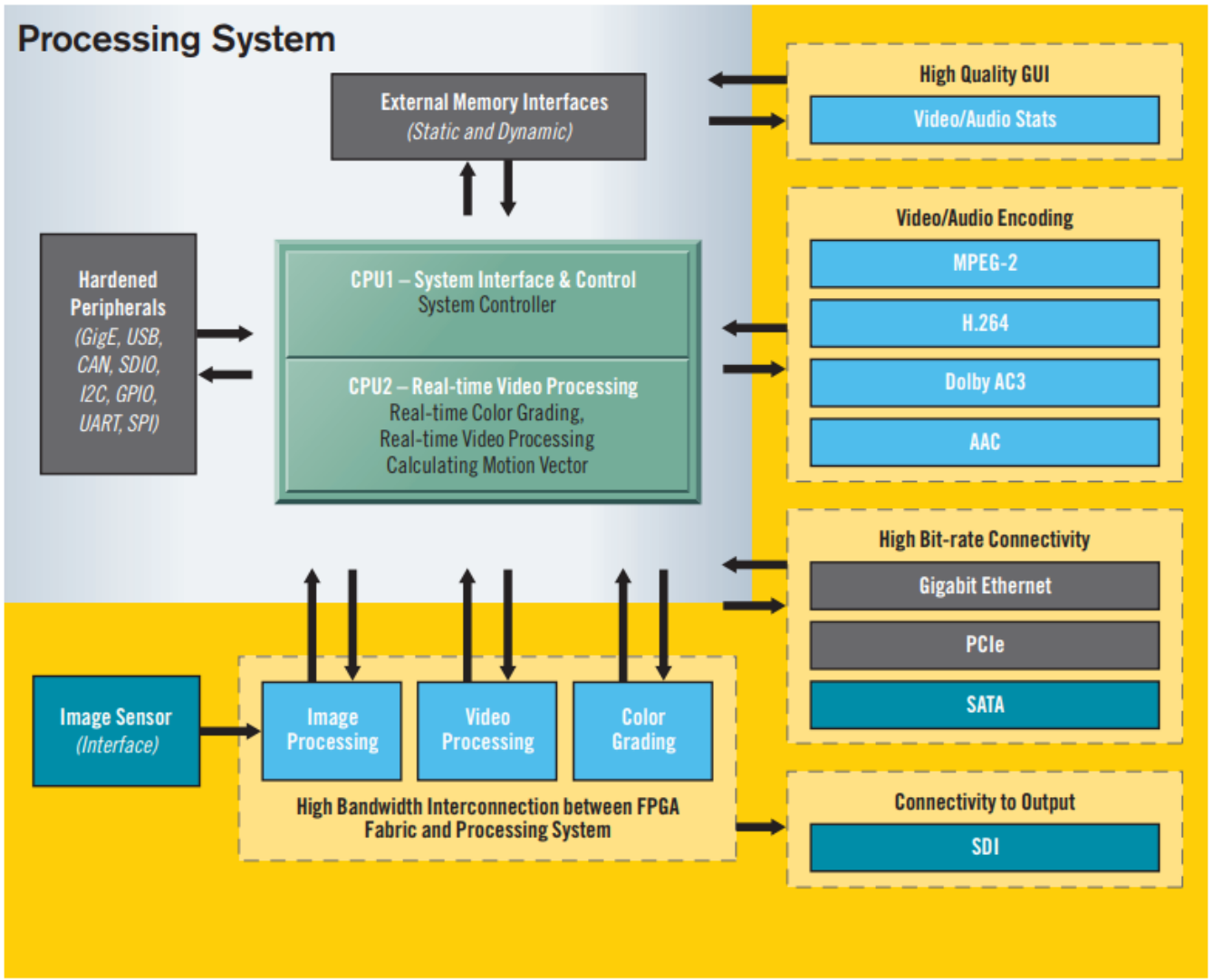
Panoradio SDR

DC9ST

Multi-Camera Multi-Feature Driver Assistance Platform



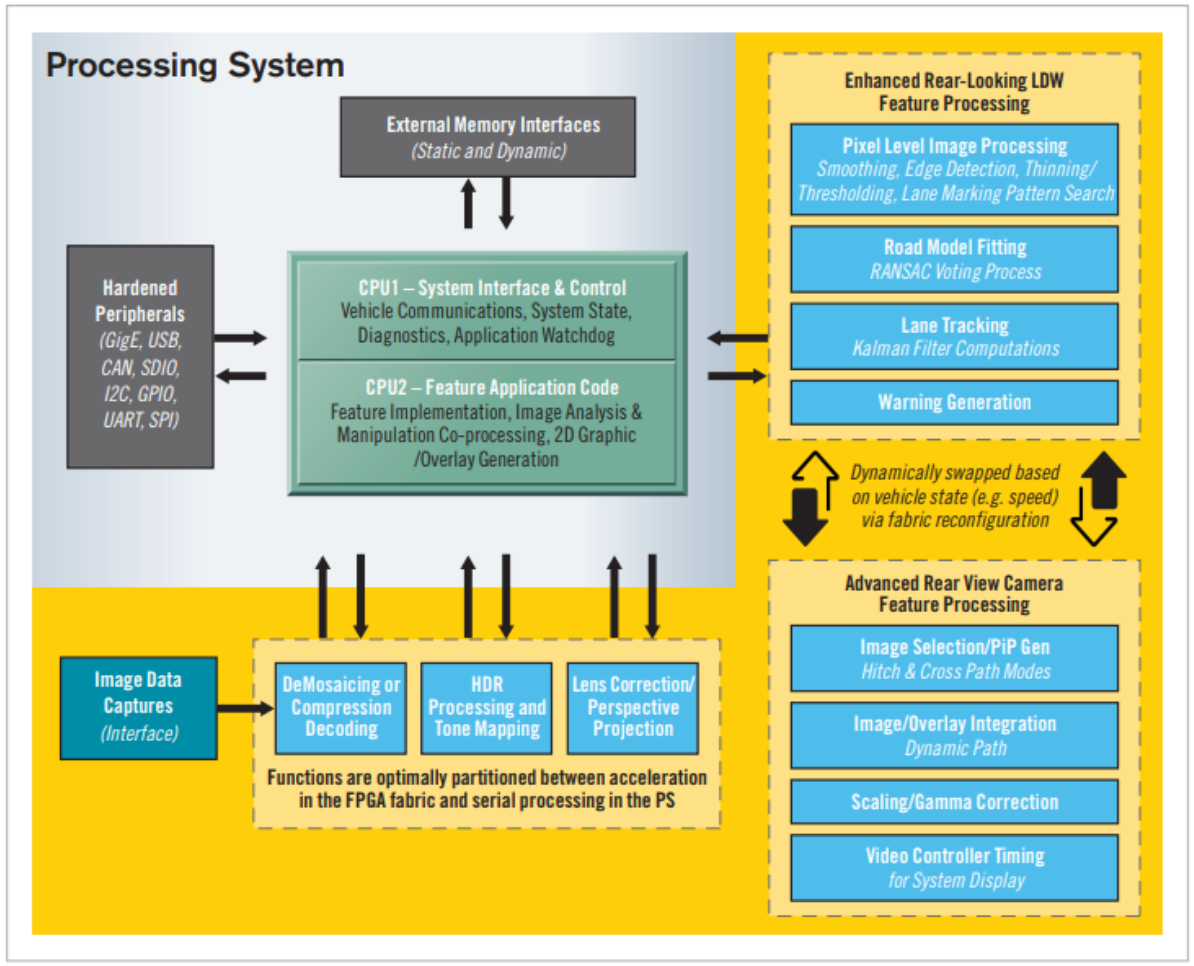
BROADCAST CAMERA APPLICATION EXAMPLE



Zynq-7000 EPP devices provide very high bit-rate bandwidth for high-accuracy video processing and analytics in broadcast-level camera systems.

- Zynq-7000 EPP devices improve time to market and accommodate for broadcasting standards and future algorithms evolution.
- The high integration in the Zynq-7000 EPP devices enables system power consumption savings and cost reduction.
- Dual ARM Cortex-A9 with NEON and dual precision offer high levels of performance for algorithmic processing and video data compression encoding (e.g. MPEG-2, H.264).
- Processing system to DSP-rich programmable logic interconnect allows high bandwidth, low latency interface to enable hardware accelerations of most video processing and video analytics functions.

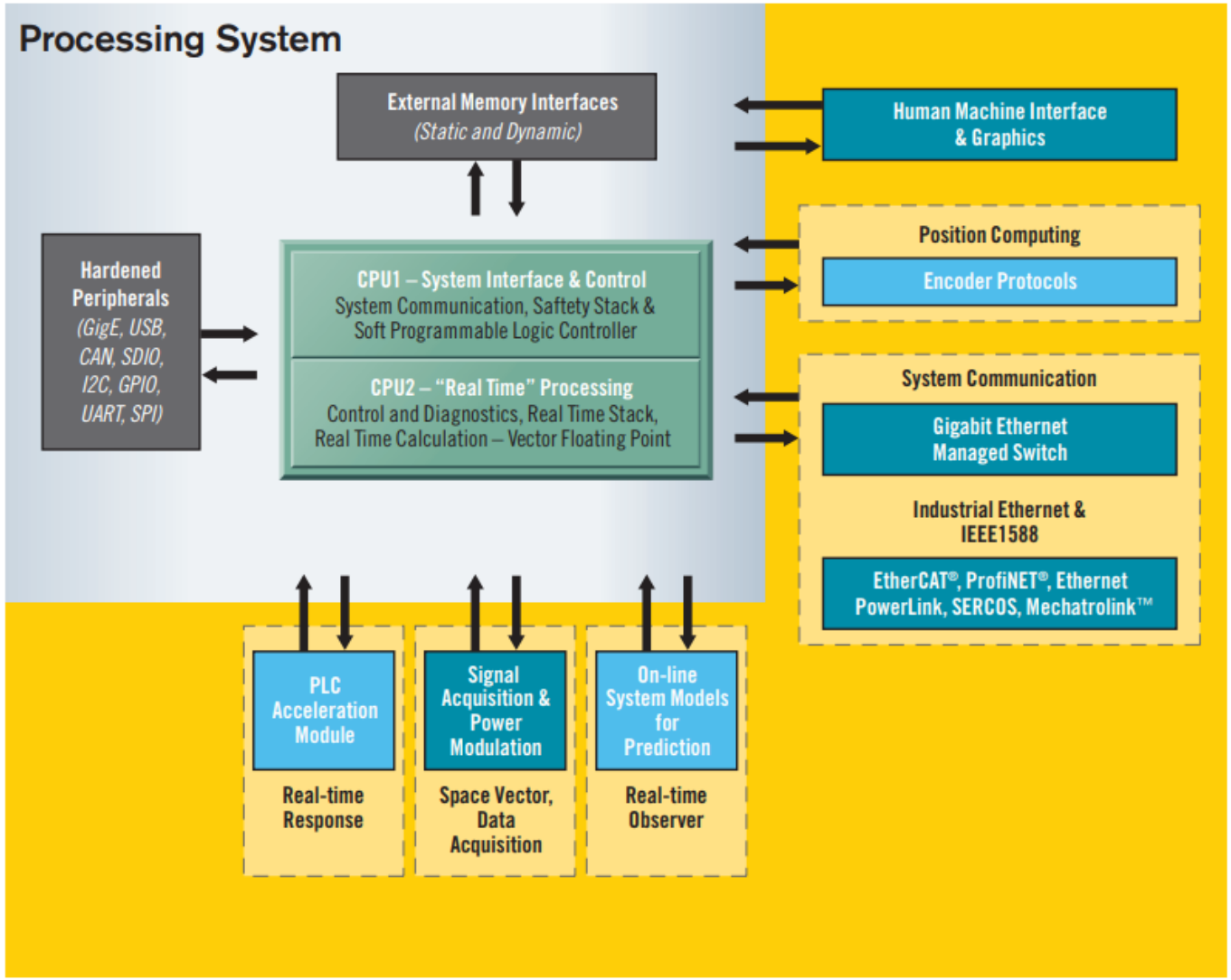
SINGLE-CAMERA / MULTI-FEATURE DRIVER ASSISTANCE APPLICATION EXAMPLE



The Zynq-7000 EPP family addresses the stringent video processing and analytics requirements for current and future driver assistance systems.

- The programmable logic accommodates emerging camera interface standards and evolving DA processing algorithms while also improving time to market.
- The tight integration provides a high bandwidth and low-latency interface for unprecedented optimization of HW/SW partitioning for computationally intensive video/image processing functions.
- Leveraging the dynamic reconfiguration capability of Zynq-7000 EPP devices allows features like rear view camera and lane departure warning systems to be dynamically swapped based on vehicle state (e.g. speed) thereby reducing total required logic and system cost.
- The scalability of the Zynq-7000 EPP family allows designers to chose between Z-7010 and Z-7020 devices and offer solutions that are ideally tailored and cost-optimized to each DA system.
- Dual Cortex-A9 processors with dual precision floating point and NEON engines offer complex algorithmic processing and support for video data compression codecs.
- Automotive compatible SoC architecture is supported by key hardened peripherals like CAN and Ethernet.

INDUSTRIAL MOTOR CONTROL APPLICATION EXAMPLE



Zynq-7000 EPP devices offer a unique blend of performance and flexibility to meet the high-processing demands and integration required for current and future motor control systems.

- Designers can accelerate time to market while enabling in-system programmability and ensuring "future proofing" of products.
- The scalability between the Z-7010 and Z-7020 EPP devices allow customers to offer bundled product solutions and support varying motor types from a single platform.
- Zynq-7000 EPP devices possess all the elements required for motor control (powerful processors, peripherals, analog-to-digital converter) which offer an integrated cost effective solution for tomorrow's greener industrial solutions.
- Tight coupling of the processing system and programmable logic enable high bandwidth low latency for real-time industrial networking interfaces and motor control hardware accelerators in a single device.