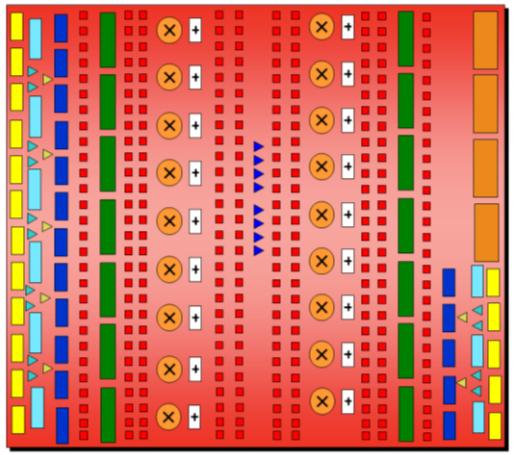
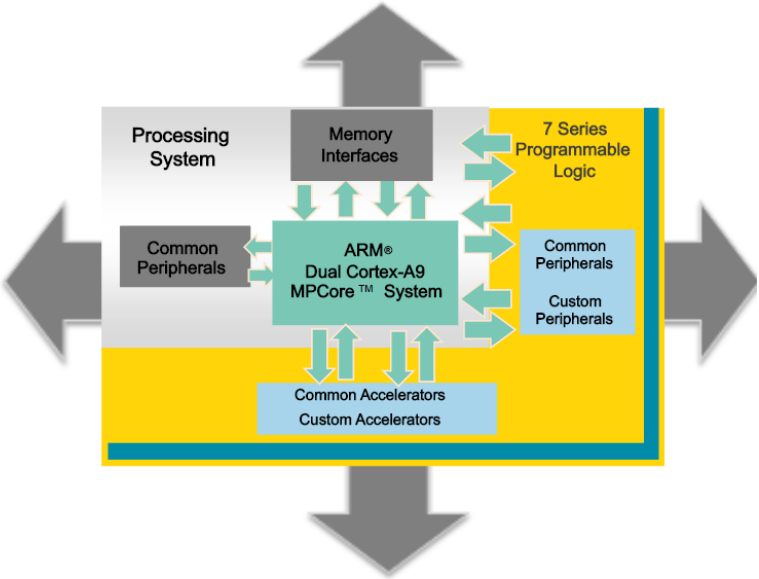
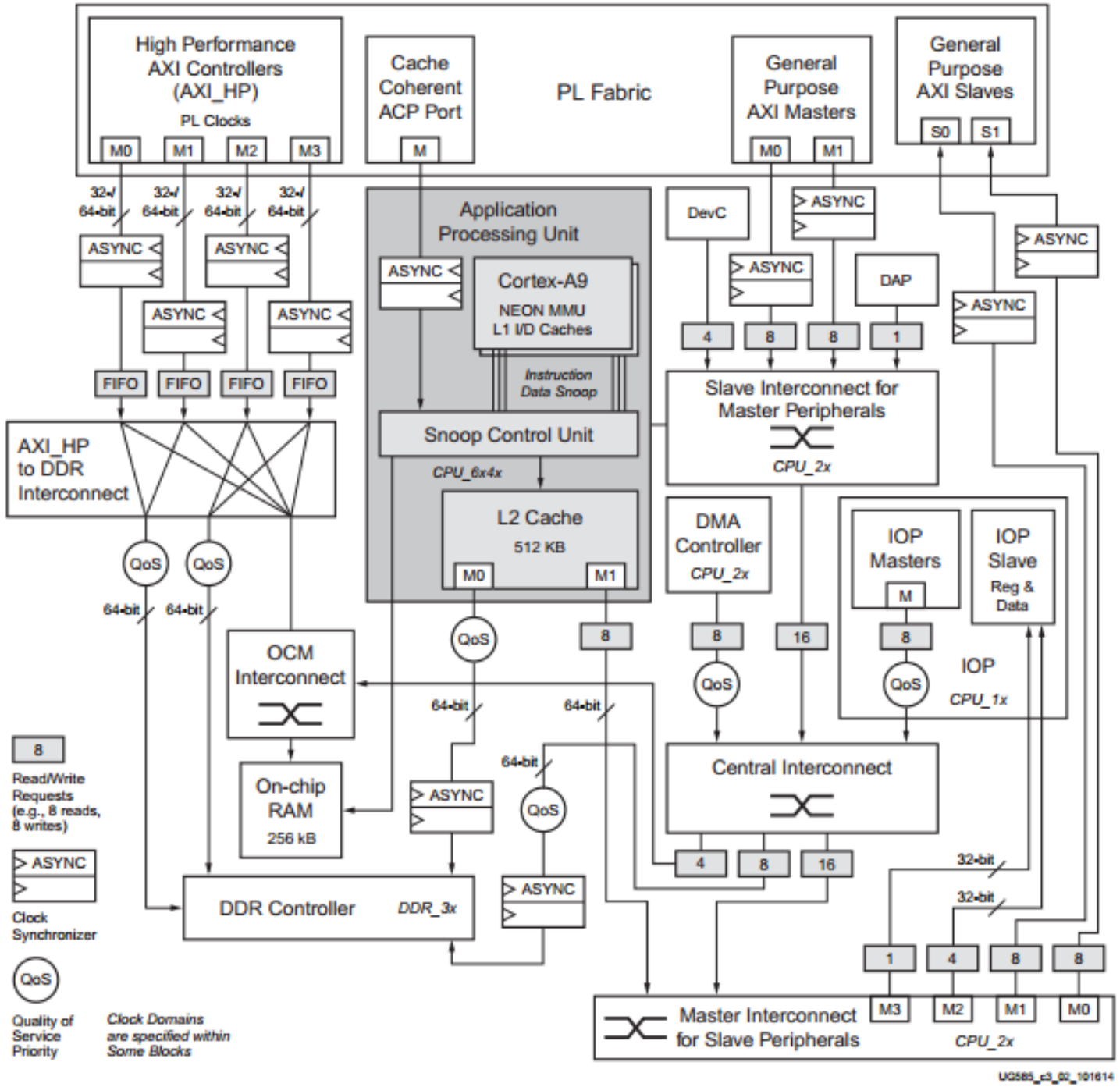
DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

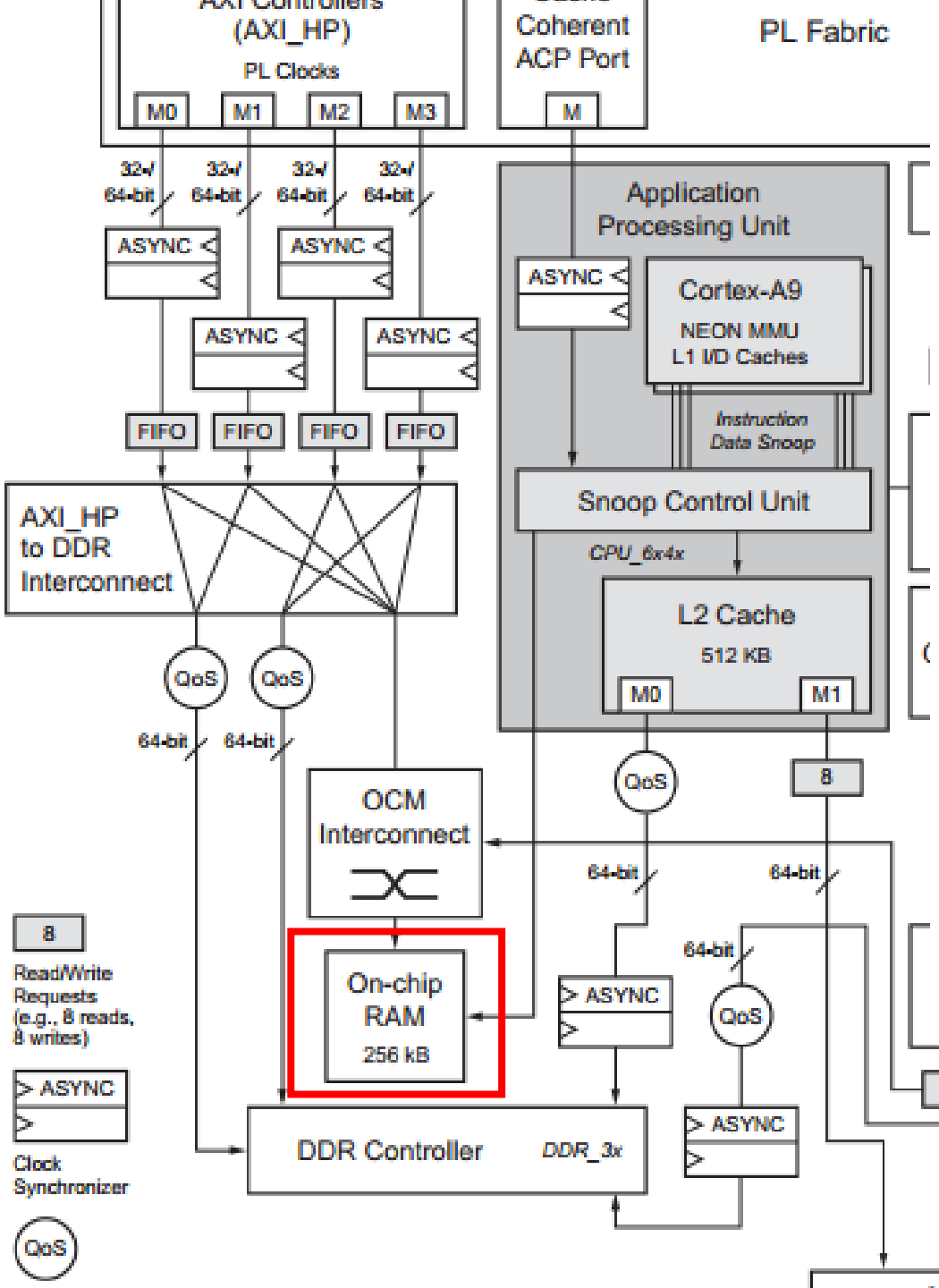
A2A
Algorithms to Architecture

ECE 270: Embedded Logic Design

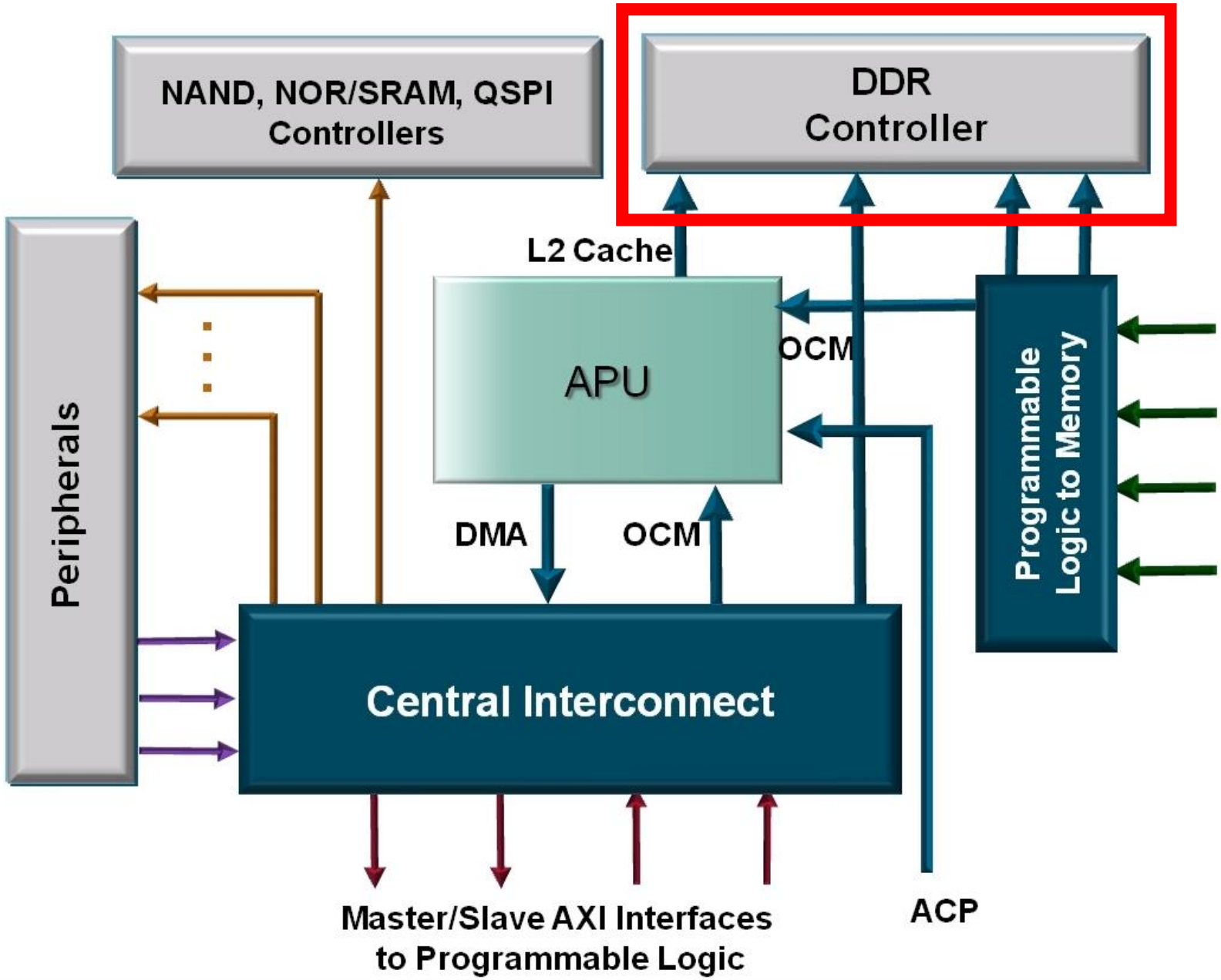


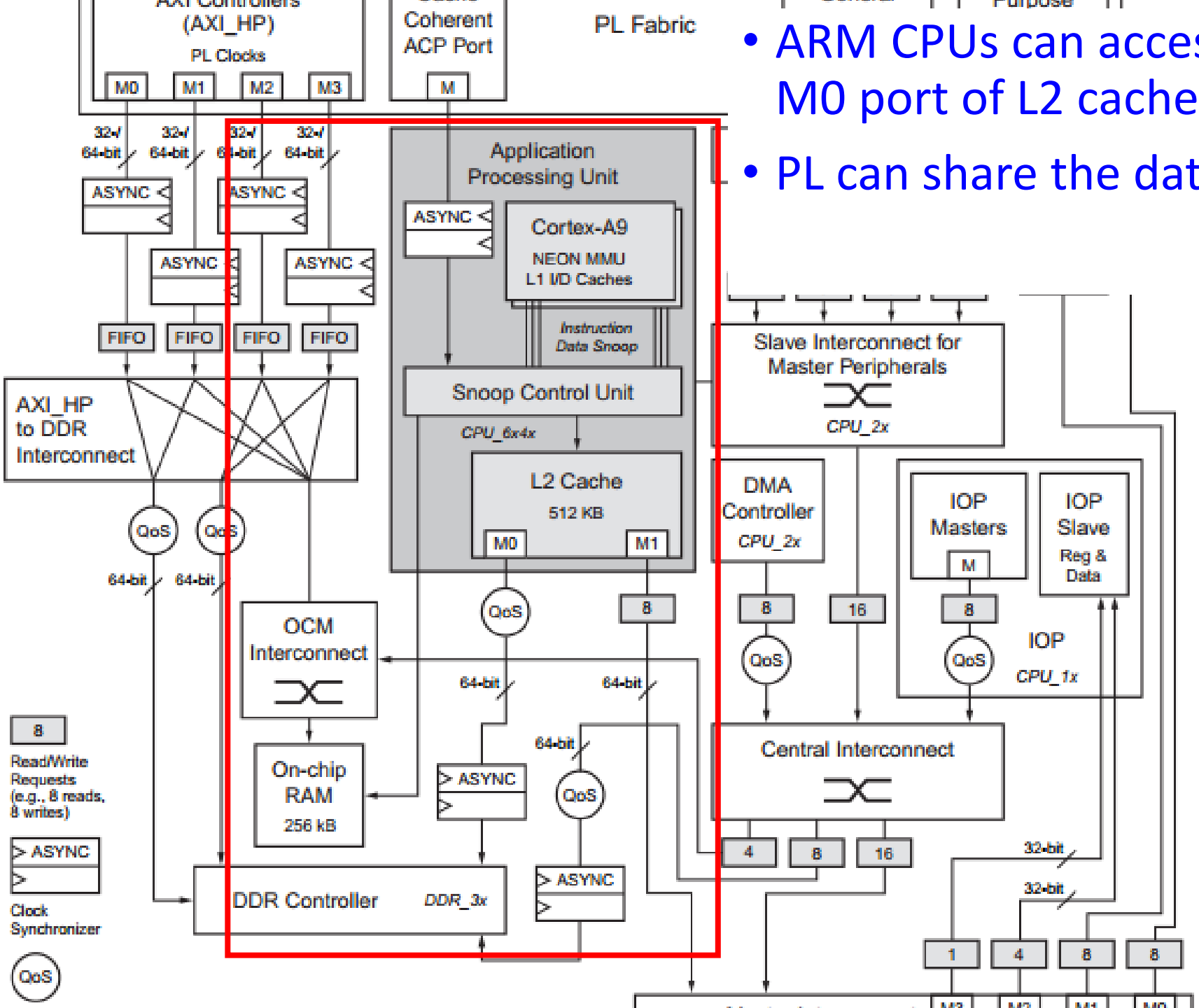



High Performance AXI Controllers (AXI_HP)
PL Clocks
M0
M1
M2
M3
Cache Coherent ACP Port
M
PL Fabric
General Purpose AXI Masters
M0
M1
General Purpose AXI Slaves
S0
S1
32-/64-bit
ASYNC
FIFO
AXI_HP to DDR Interconnect
QoS
64-bit
Application Processing Unit
Cortex-A9
NEON MMU
L1 I/D Caches
Instruction
Data Snoop
Snoop Control Unit
CPU_6x4x
L2 Cache
512 KB
DevC
DAP
4
8
1
Slave Interconnect for Master Peripherals
CPU_2x
DMA Controller
CPU_2x
16
IOP Masters
M
IOP Slave
Reg & Data
IOP
CPU_1x
OCM Interconnect
On-chip RAM
256 kB
Central Interconnect
DDR Controller
DDR_3x
32-bit
Master Interconnect for Slave Peripherals
CPU_2x
M3
M2
M1
M0
Read/Write Requests (e.g., 8 reads, 8 writes)
Clock Synchronizer
Quality of Service Priority
Clock Domains are specified within Some Blocks
UG585_c3_02_101614

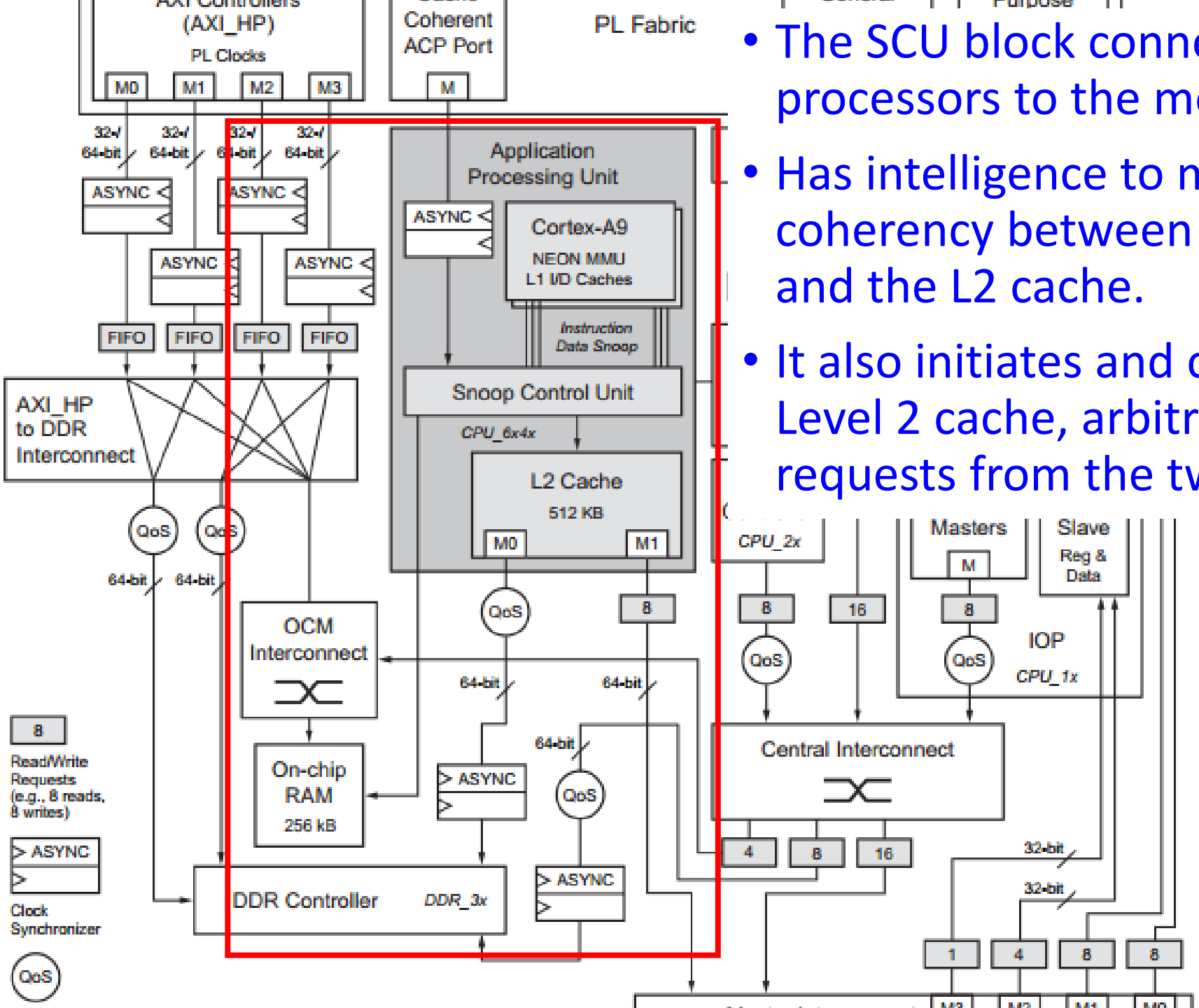


- The on-chip memory (OCM) module contains 256 KB of RAM and 128 KB of ROM (BootROM).
- The BootROM memory is used exclusively by the boot process and is not visible to the user.
- OCM supports two 64-bit AXI slave interface ports.
- One dedicated for CPU/ACP access via the APU snoop control unit (SCU)
- Other shared by all other bus masters within the processing system (PS) and programmable logic (PL).

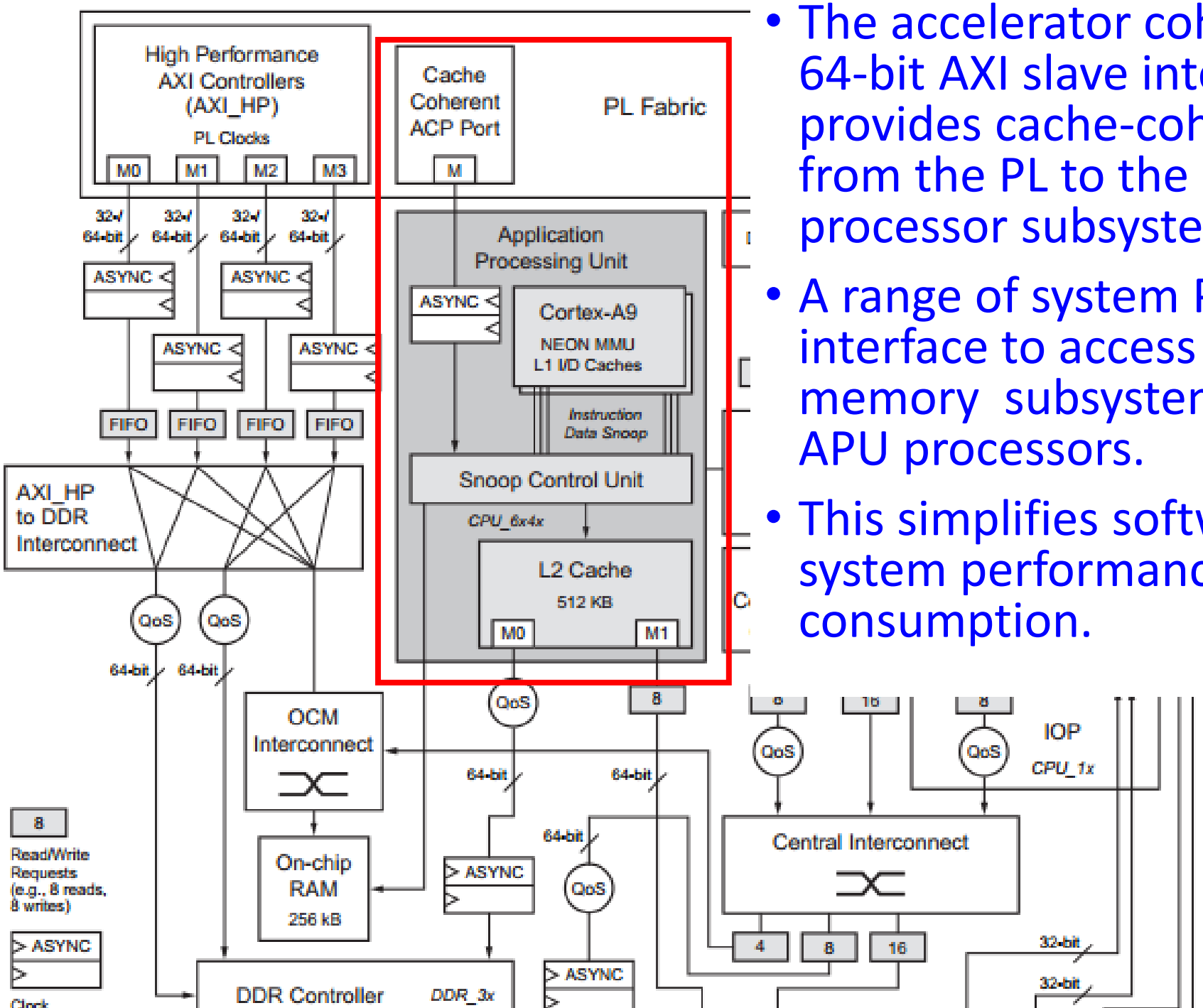

NAND, NOR/SRAM, QSPI Controllers
DDR Controller
L2 Cache
APU
OCM
Programmable Logic to Memory
Peripherals
DMA
OCM
Central Interconnect
Master/Slave AXI Interfaces to Programmable Logic
ACP



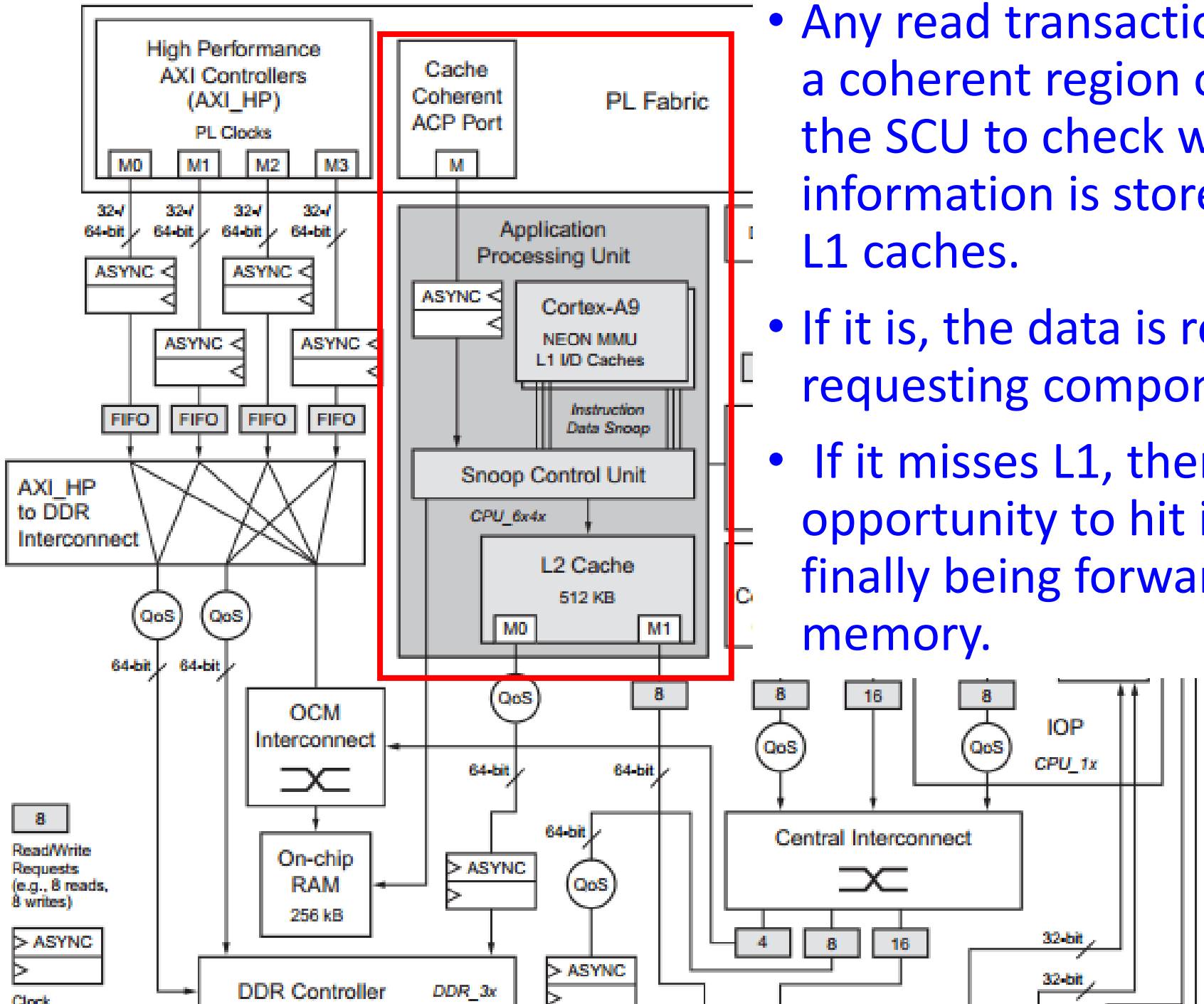
- ARM CPUs can access DDR memory via M0 port of L2 cache and SCU
- PL can share the data with CPU via DDR.



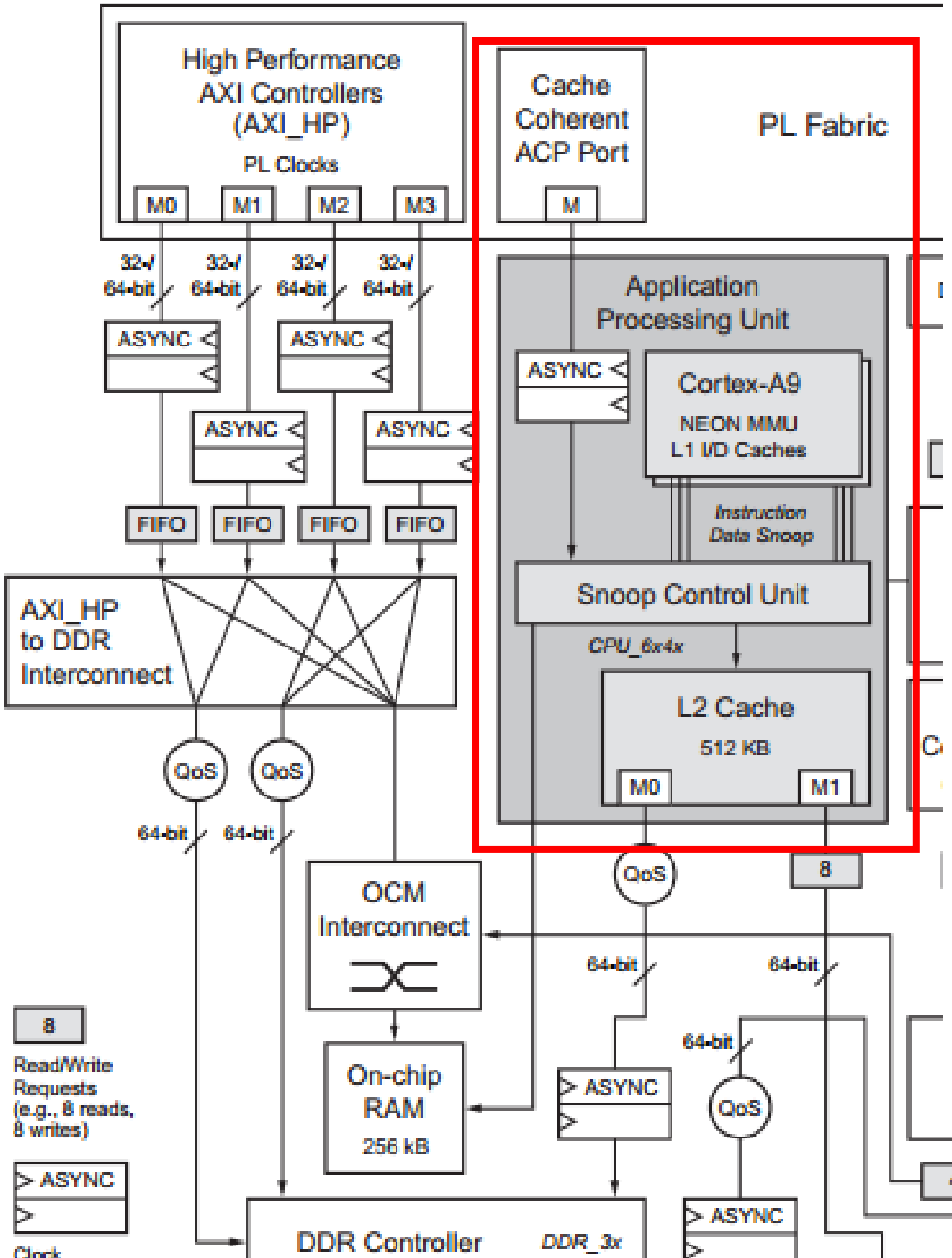
- The SCU block connects the two Cortex-A9 processors to the memory subsystem
- Has intelligence to manage the data cache coherency between the two processors and the L2 cache.
- It also initiates and controls access to the Level 2 cache, arbitrating between requests from the two cores



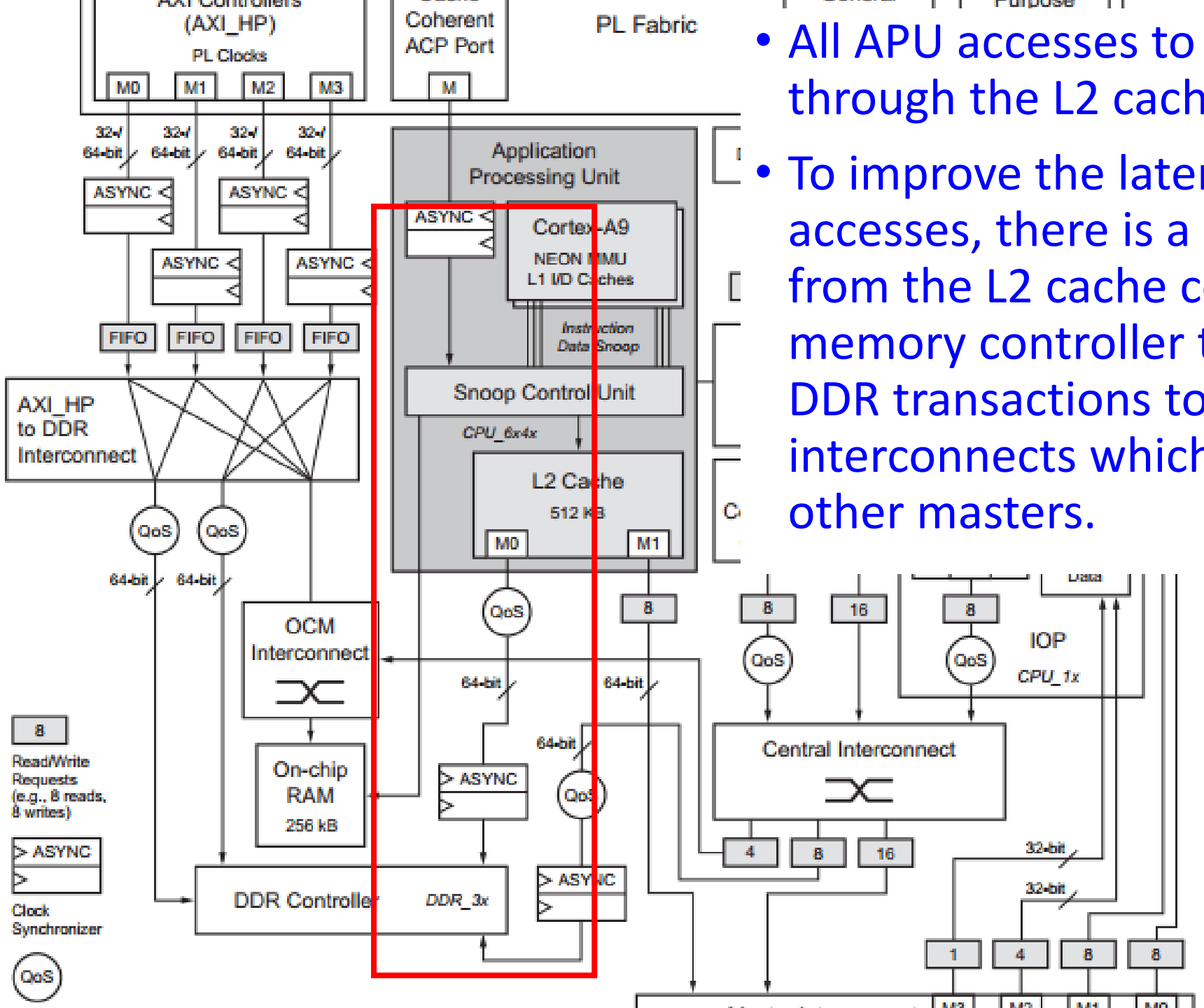
- The accelerator coherency port (ACP) is a 64-bit AXI slave interface on the SCU that provides cache-coherent access directly from the PL to the Cortex-A9 MP-Core processor subsystem.
- A range of system PL masters can use this interface to access the caches and the memory subsystem exactly the way the APU processors.
- This simplifies software, increase overall system performance, or improve power consumption.



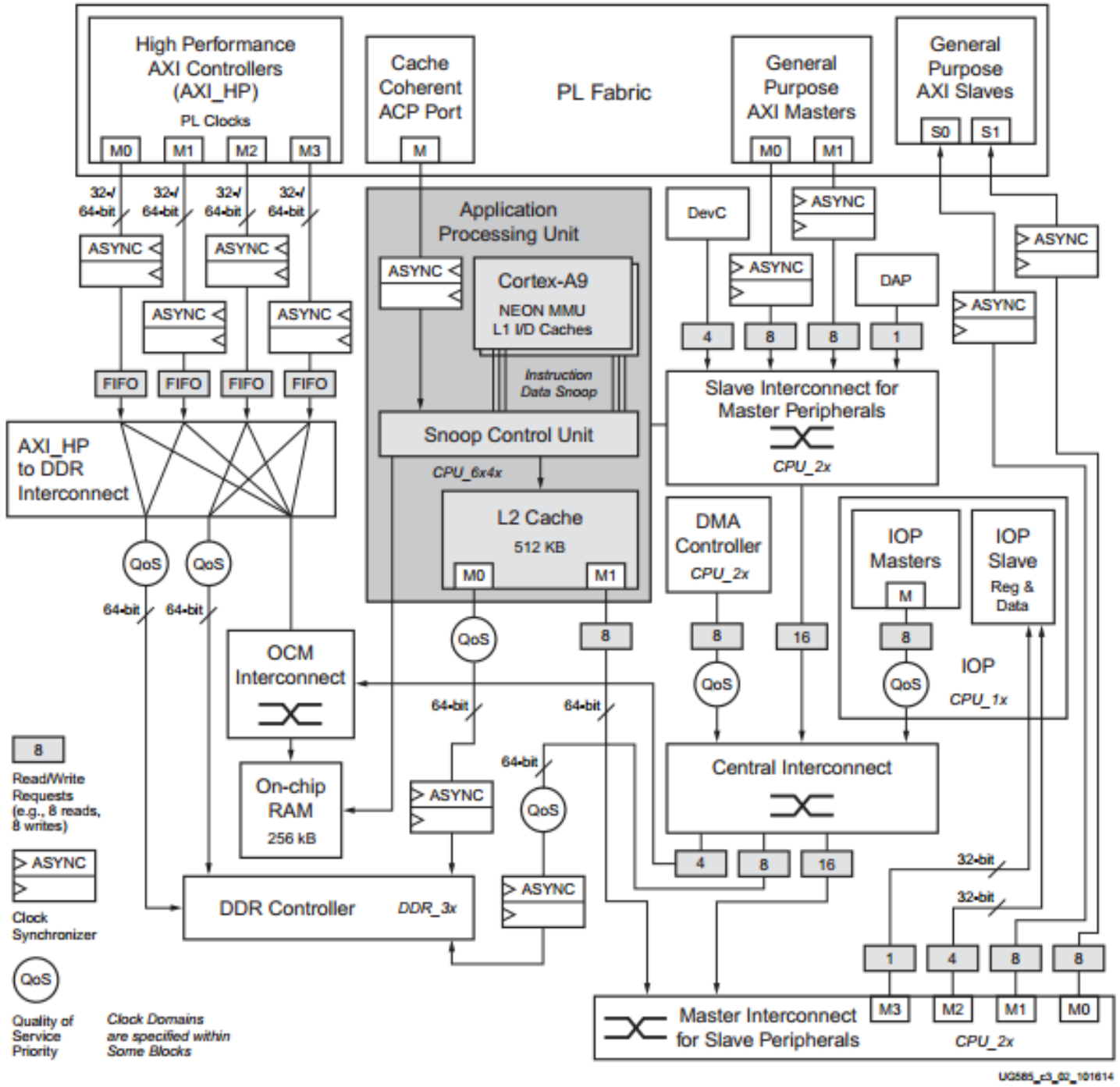
- Any read transactions through the ACP to a coherent region of memory interact with the SCU to check whether the required information is stored within the processor L1 caches.
- If it is, the data is returned directly to the requesting component.
- If it misses L1, then there is also the opportunity to hit in L2 cache before finally being forwarded to the main memory.

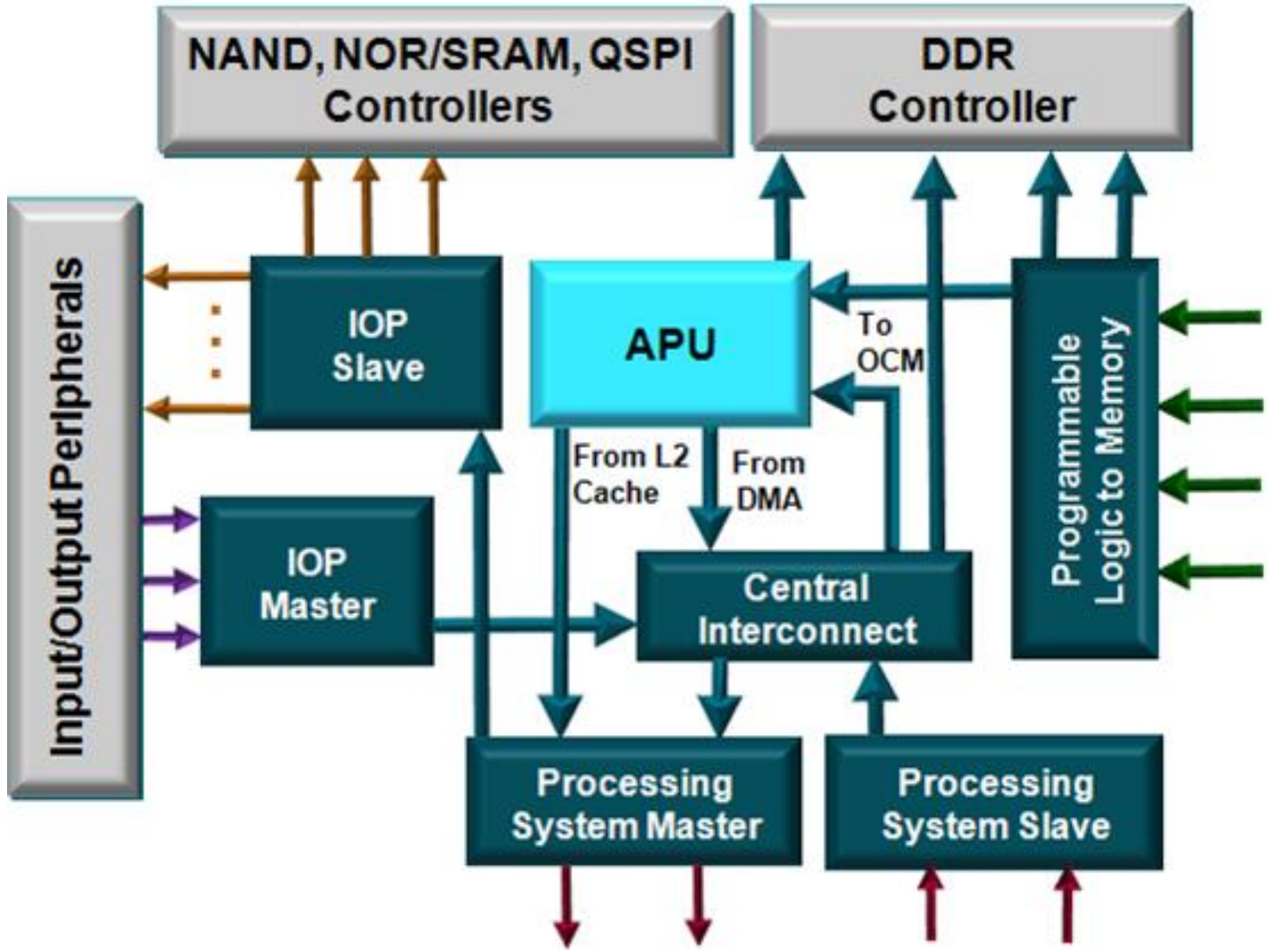


- The ACP provides a low latency path between the PS and the accelerators implemented in the PL when compared with a legacy cache flushing and loading scheme.
- AXI master on PL can also access OCM via SNU

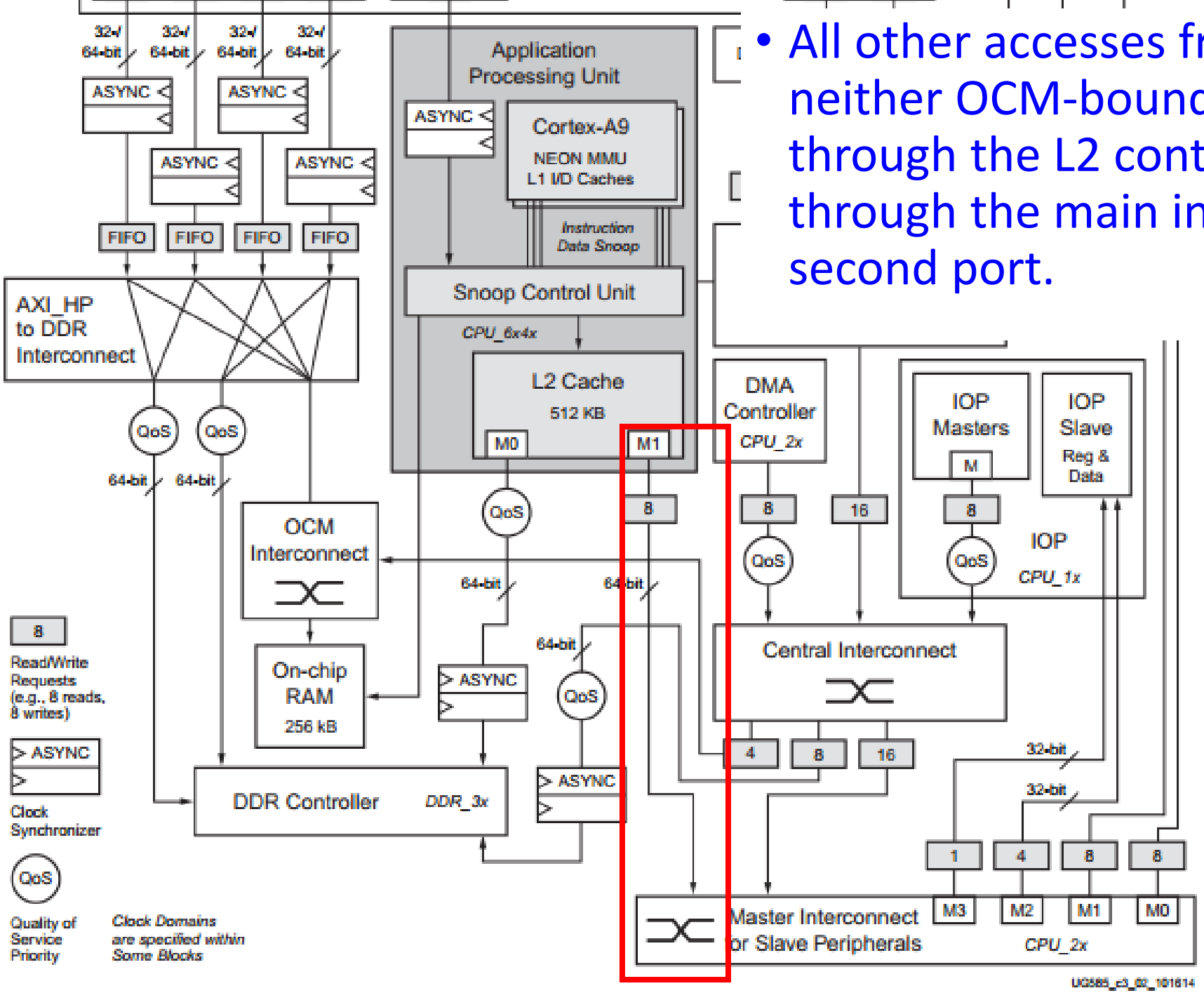


- All APU accesses to the DDR are routed through the L2 cache controller.
- To improve the latencies of the DDR accesses, there is a dedicated master port from the L2 cache controller to the DDR memory controller that allows all APU-DDR transactions to bypass the main interconnects which are shared with the other masters.

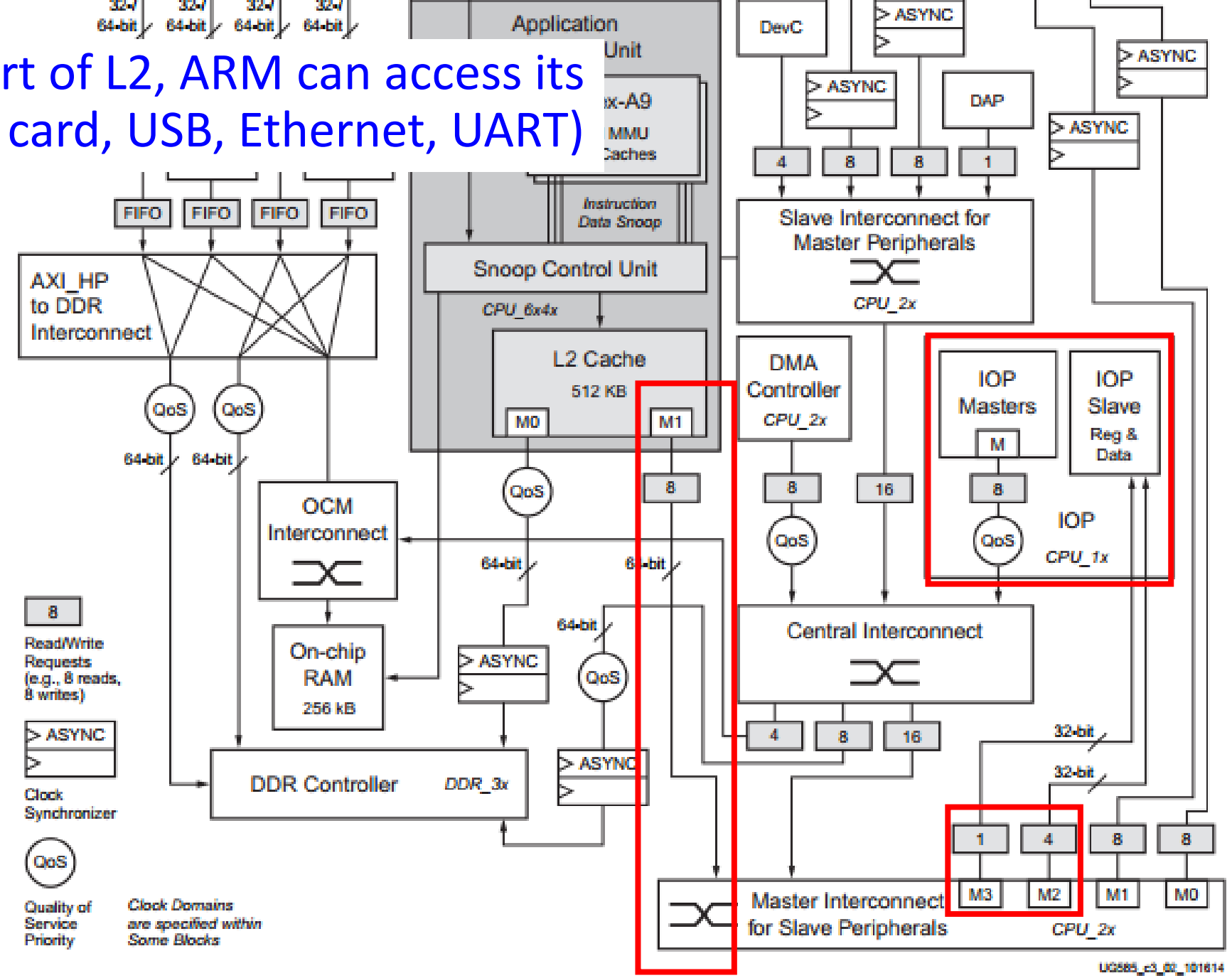

High Performance AXI Controllers (AXI_HP)
PL Clocks
M0
M1
M2
M3
Cache Coherent ACP Port
M
PL Fabric
General Purpose AXI Masters
M0
M1
General Purpose AXI Slaves
S0
S1
32-/64-bit
ASYNC
FIFO
AXI_HP to DDR Interconnect
QoS
64-bit
Application Processing Unit
Cortex-A9
NEON MMU
L1 I/D Caches
Instruction
Data Snoop
Snoop Control Unit
CPU_6x4x
L2 Cache
512 KB
DevC
DAP
4
8
1
Slave Interconnect for Master Peripherals
CPU_2x
DMA Controller
CPU_2x
16
IOP Masters
M
IOP Slave
Reg & Data
IOP
CPU_1x
OCM Interconnect
On-chip RAM
256 kB
Central Interconnect
DDR Controller
DDR_3x
32-bit
Master Interconnect for Slave Peripherals
CPU_2x
M3
M2
M1
M0
Read/Write Requests (e.g., 8 reads, 8 writes)
Clock Synchronizer
Quality of Service Priority
Clock Domains are specified within Some Blocks
UG585_c3_02_101614

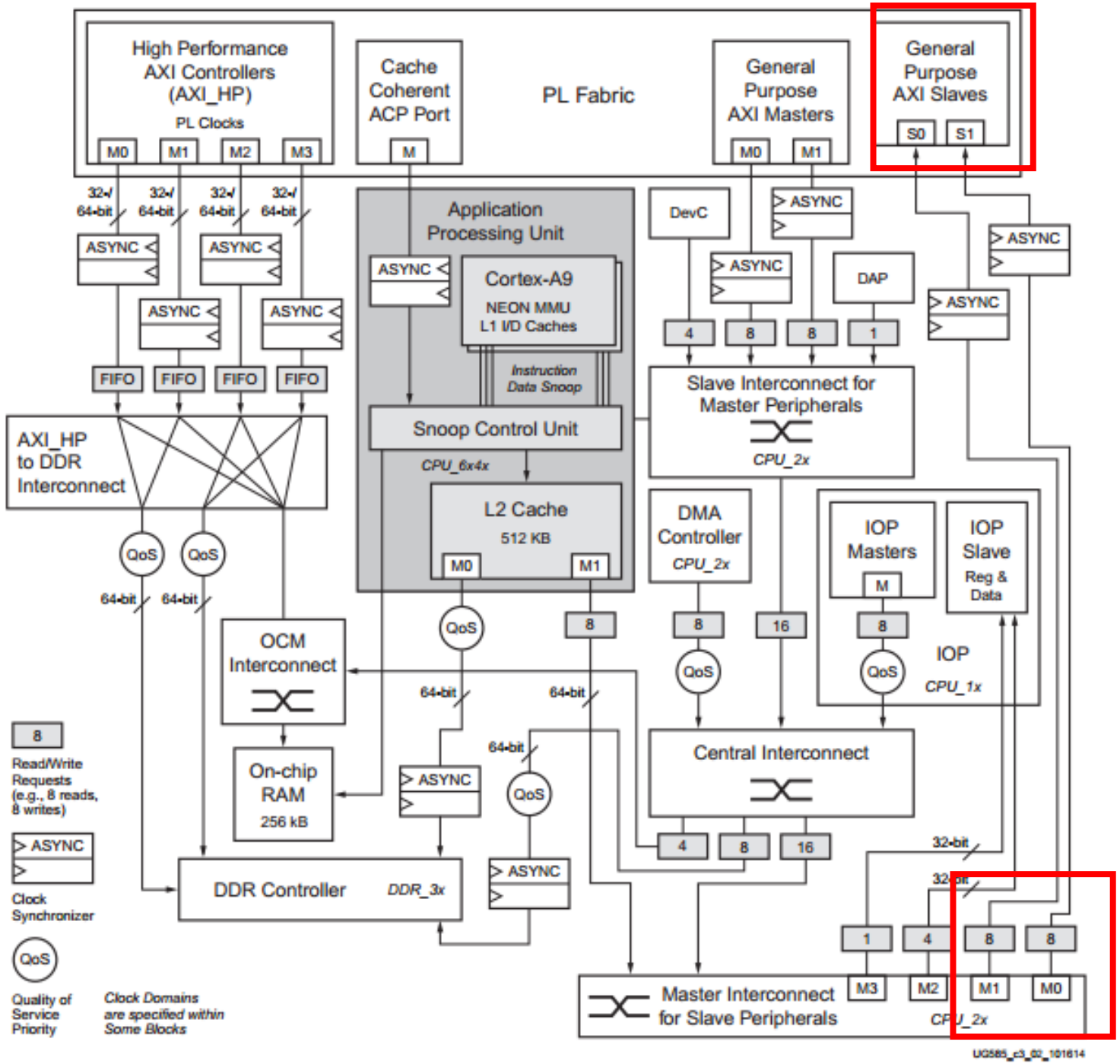

NAND, NOR/SRAM, QSPI Controllers
DDR Controller
Input/Output Perlpherals
IOP Slave
APU
To OCM
Programmable Logic to Memory
From L2 Cache
From DMA
IOP Master
Central Interconnect
Processing System Master
Processing System Slave

- All other accesses from the APU that are neither OCM-bound nor DDR-bound go through the L2 controller and are routed through the main interconnect using a second port.

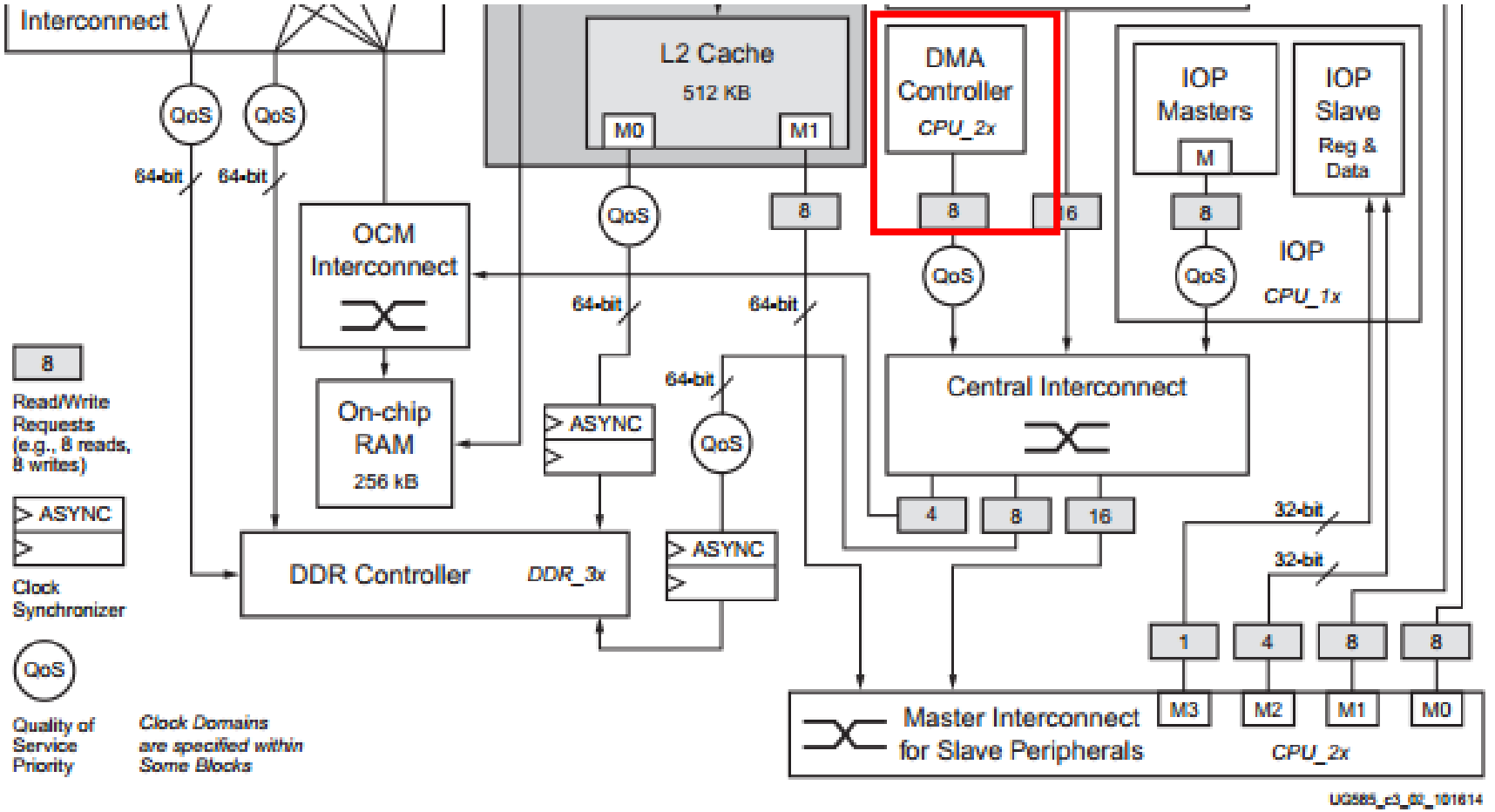


- Through M1 port of L2, ARM can access its peripherals (SD card, USB, Ethernet, UART)



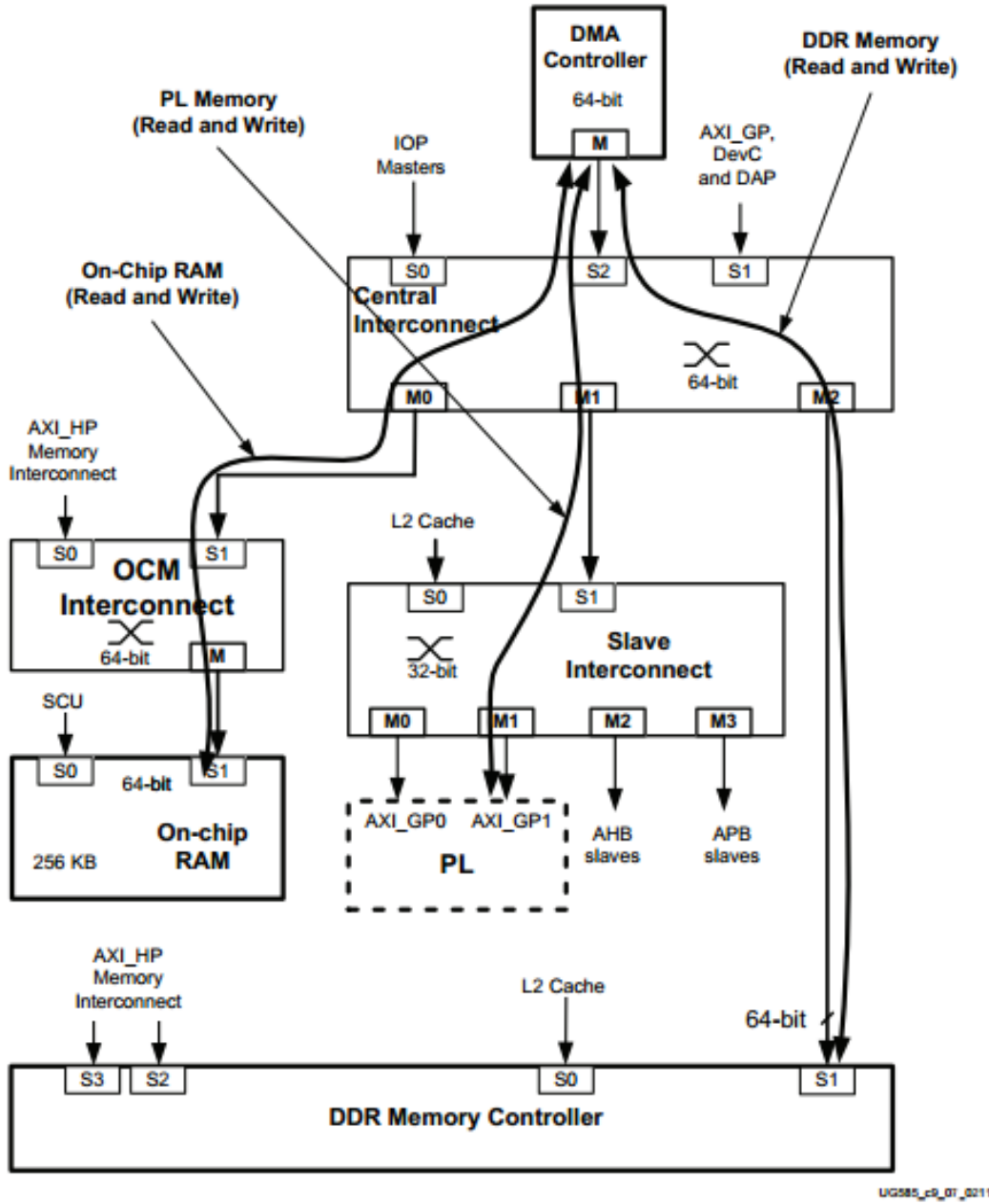

High Performance AXI Controllers (AXI_HP)
PL Clocks
M0
M1
M2
M3
Cache Coherent ACP Port
M
PL Fabric
General Purpose AXI Masters
M0
M1
General Purpose AXI Slaves
S0
S1
32-/64-bit
ASYNC
FIFO
AXI_HP to DDR Interconnect
QoS
64-bit
Application Processing Unit
Cortex-A9
NEON MMU
L1 I/D Caches
Instruction
Data Snoop
Snoop Control Unit
CPU_6x4x
L2 Cache
512 KB
DevC
DAP
4
8
1
Slave Interconnect for Master Peripherals
CPU_2x
DMA Controller
CPU_2x
16
IOP Masters
IOP Slave
Reg & Data
IOP
CPU_1x
OCM Interconnect
On-chip RAM
256 kB
Central Interconnect
DDR Controller
DDR_3x
32-bit
Master Interconnect for Slave Peripherals
CPU_2x
8
Read/Write Requests (e.g., 8 reads, 8 writes)
Clock Synchronizer
Quality of Service Priority
Clock Domains are specified within Some Blocks
UG585_c3_02_101614

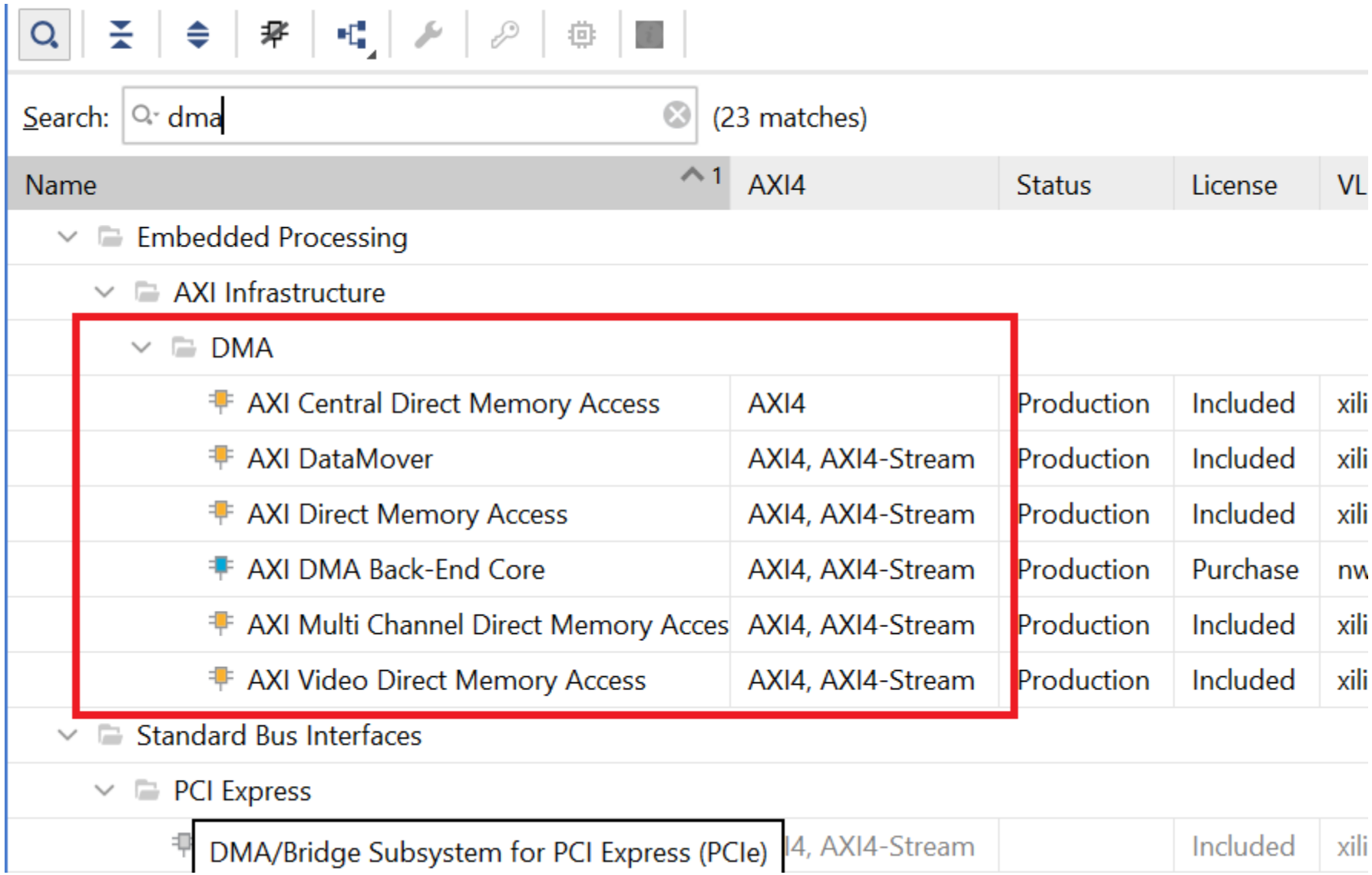
- DMA controller is able to move large amounts of data without processor intervention.
- It can be programmed by CPU.
- It is master for central interconnect which is connected to DDR, OCM and slave peripherals including PL



- Two typical DMA transaction examples include:

1. Memory to memory (On-chip memory to DDR memory)
2. Memory to/from PL/PS peripheral (DDR memory to PL/PS peripheral)




Search: dma
(23 matches)
Name
AXI4
Status
License
VL
Embedded Processing
AXI Infrastructure
DMA
AXI Central Direct Memory Access
AXI4
Production
Included
xili
AXI DataMover
AXI4, AXI4-Stream
Production
Included
xili
AXI Direct Memory Access
AXI4, AXI4-Stream
Production
Included
xili
AXI DMA Back-End Core
AXI4, AXI4-Stream
Production
Purchase
nw
AXI Multi Channel Direct Memory Acces
AXI4, AXI4-Stream
Production
Included
xili
AXI Video Direct Memory Access
AXI4, AXI4-Stream
Production
Included
xili
Standard Bus Interfaces
PCI Express
DMA/Bridge Subsystem for PCI Express (PCIe)
Included
xili

Various DMAs in PL

- **AXI CDMA**
- Provides high-bandwidth Direct Memory Access (DMA) between a memory-mapped source address and a memory-mapped destination address using the AXI4 protocol.
- If your BRAM is already hanging off the AXI interconnect, this will probably be the best option
- Optional Scatter Gather (SG) feature

Various DMAs in PL

- **AXI Datamover** - Basic AXI4 Read to AXI4-Stream and AXI4-Stream to AXI4 Write data transport and protocol conversion.
- Enables high throughput transfer of data between the AXI4 memory-mapped and AXI4-Stream domains
- Lower level control of commands than other DMA (No interrupts)

Various DMAs in PL

- **AXI DMA** - AXI Stream to AXI memory-mapped transfers
- **AXI VDMA** - Similar to AXI DMA, but it does 2D transfers and has some other video-specific features (probably not what you want, unless you're doing video/imaging)

AXI DMA: Configuration by CPU and Polling

- Remove printf statements from timing calculation

```
// Simple DMA Transfers
status=XAxiDma_SimpleTransfer(&AxiDMA,(UINTPTR)FFT_output_hw,(sizeof(float complex)*N),XAXIDMA_DEVICE_TO_DMA);
status=XAxiDma_SimpleTransfer(&AxiDMA,(UINTPTR)FFT_input,(sizeof(float complex)*N),XAXIDMA_DMA_TO_DEVICE);

// POLLING-Check whether the DMA-to-Device and Device-to-DMA transfers are complete
while(XAxiDma_Busy(&AxiDMA,XAXIDMA_DMA_TO_DEVICE));
printf("\n\rDMA-to-Device Transfer Done!");
while(XAxiDma_Busy(&AxiDMA,XAXIDMA_DEVICE_TO_DMA));
printf("\n\rDevice-to-DMA Transfer Done!");
XTime_GetTime(&PL_end_time);// Get End Time
```

AXI Direct Memory Access (7.1)

Documentation | IP Location

Show disabled ports

Component Name: axi_dma_0

- [ ] Enable Asynchronous Clocks (Auto)
- [ ] Enable Scatter Gather Engine
- [ ] Enable Micro DMA
- [ ] Enable Multi Channel Support
- [ ] Enable Control / Status Stream

Width of Buffer Length Register (8-26): 23 bits

Address Width (32-64): 32 bits

Enable Read Channel

Number of Channels	1
Memory Map Data Width	64
Stream Data Width	64
Max Burst Size	32

- [x] Allow Unaligned Transfers

Enable Write Channel

Number of Channels	1
MANUAL Memory Map Data Width	64
Stream Data Width (Auto)	32
Max Burst Size	32

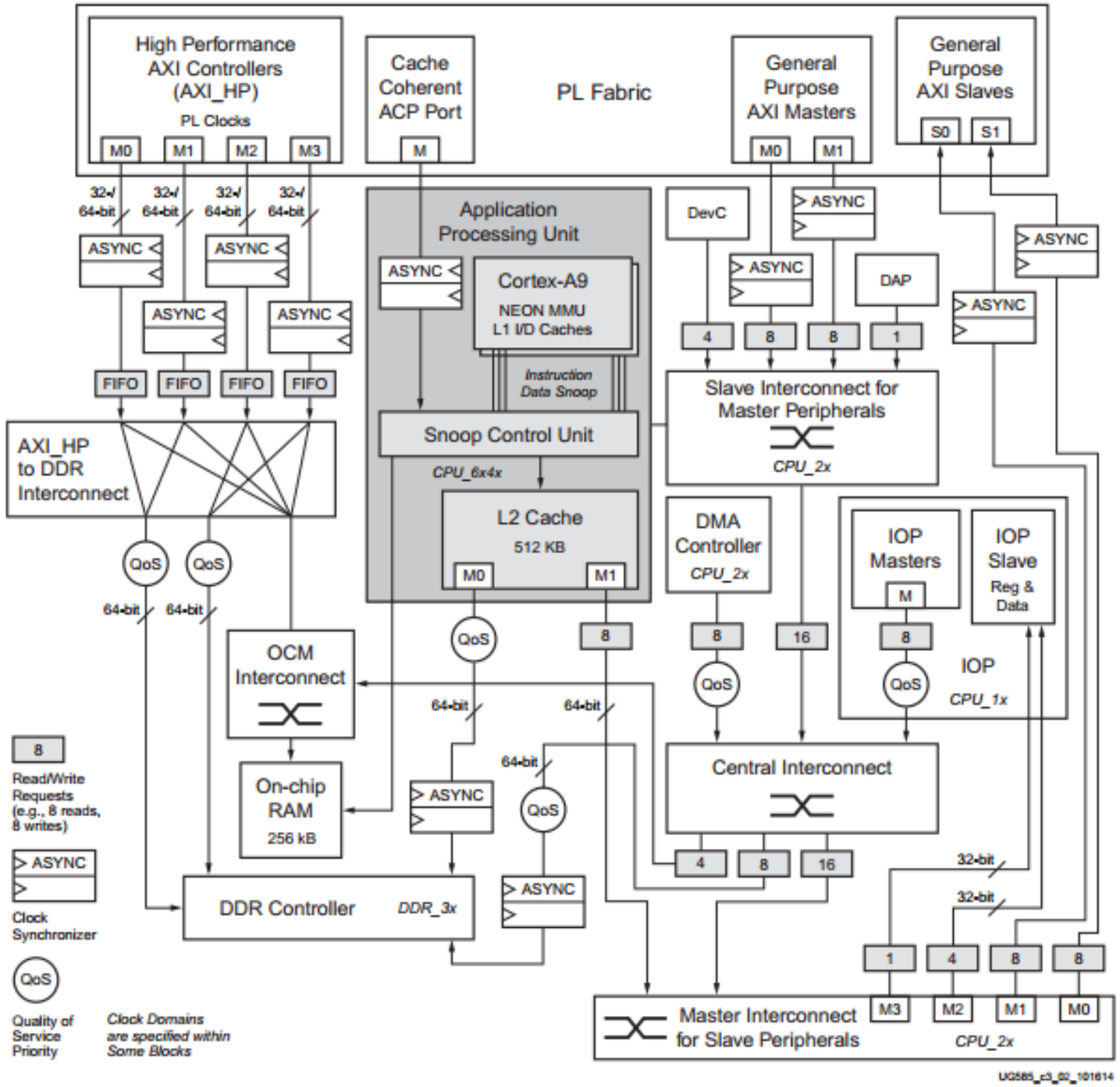
- [x] Allow Unaligned Transfers
- [ ] Use Rxlength In Status Stream

AUTO

- [ ] Enable Single AXI4 Data Interface

OK | Cancel

Lab Extensions


High Performance AXI Controllers (AXI_HP)
PL Clocks
M0
M1
M2
M3
Cache Coherent ACP Port
M
PL Fabric
General Purpose AXI Masters
M0
M1
General Purpose AXI Slaves
S0
S1
32-/64-bit
ASYNC
FIFO
AXI_HP to DDR Interconnect
QoS
64-bit
Application Processing Unit
Cortex-A9
NEON MMU
L1 I/D Caches
Instruction
Data Snoop
Snoop Control Unit
CPU_6x4x
L2 Cache
512 KB
DevC
DAP
4
8
1
Slave Interconnect for Master Peripherals
CPU_2x
DMA Controller
CPU_2x
16
IOP Masters
M
IOP Slave
Reg & Data
IOP
CPU_1x
OCM Interconnect
On-chip RAM
256 kB
Central Interconnect
DDR Controller
DDR_3x
32-bit
Master Interconnect for Slave Peripherals
CPU_2x
M3
M2
M1
M0
Read/Write Requests (e.g., 8 reads, 8 writes)
Clock Synchronizer
Quality of Service Priority
Clock Domains are specified within Some Blocks
UG585_c3_02_101614

Zynq Architecture: PS and PL