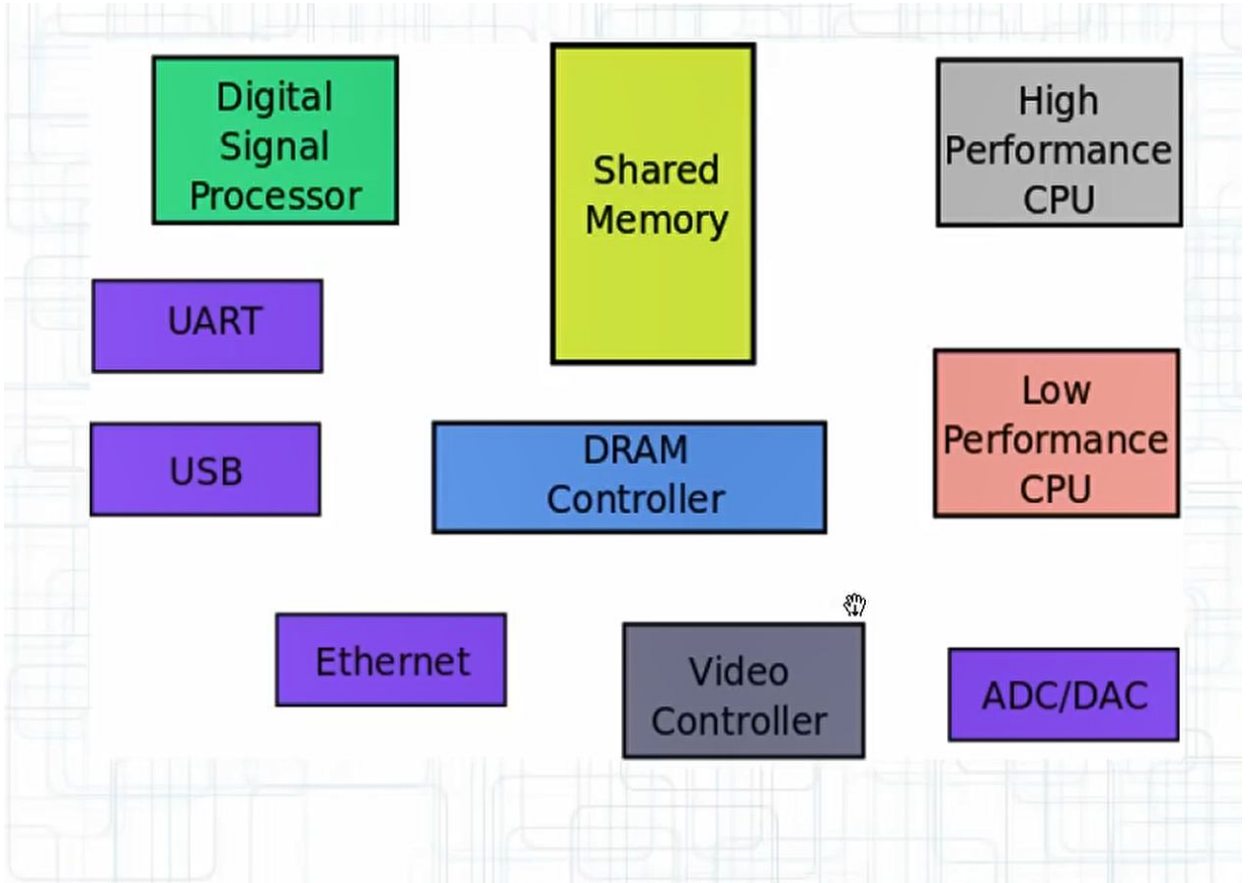
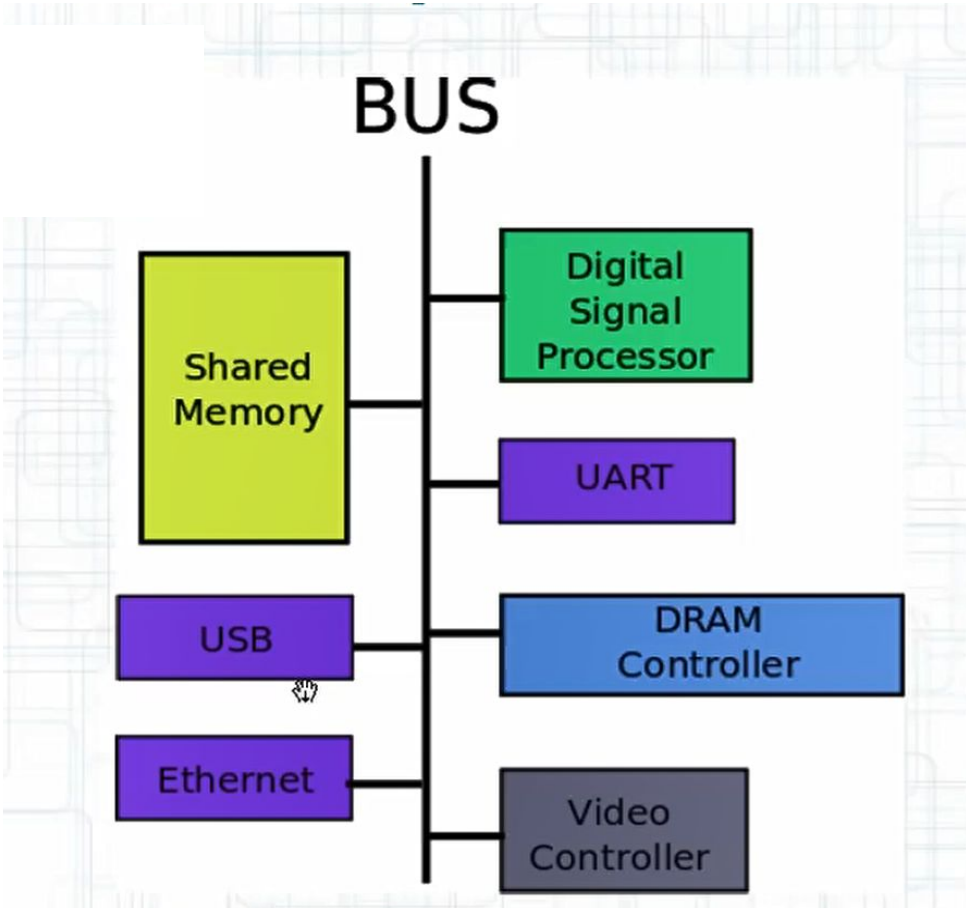
Advanced eXtensible Interface(AXI)





In any chip we some modules like this.Having modules is not sufficient they should able to communicate, to do this we should connect a bus which transfer data from one point to another on chip.

Advantages of connecting buses
1)makes standard(organized) communication so that communicate easily
2)makes easy to debug
3)makes easy to reuse in other designs by creating IP

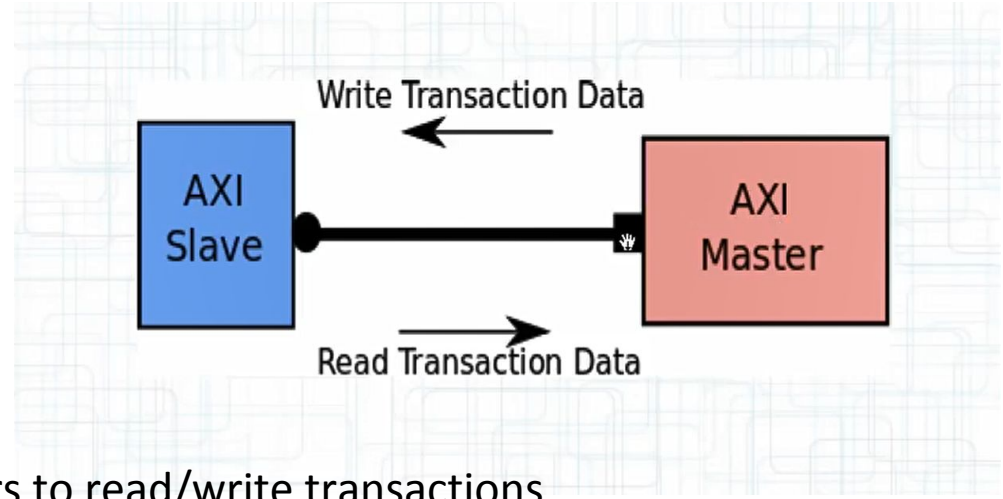
There are many SOC buses like Core Connect,Wish Bone,AXI..
In these AXI is extensively used which is designed by ARM.Vivado design flow is based on AXI interfaces.

AXI is a bus or this is a kind of protocall or way of signalling through which modules can communicate with others

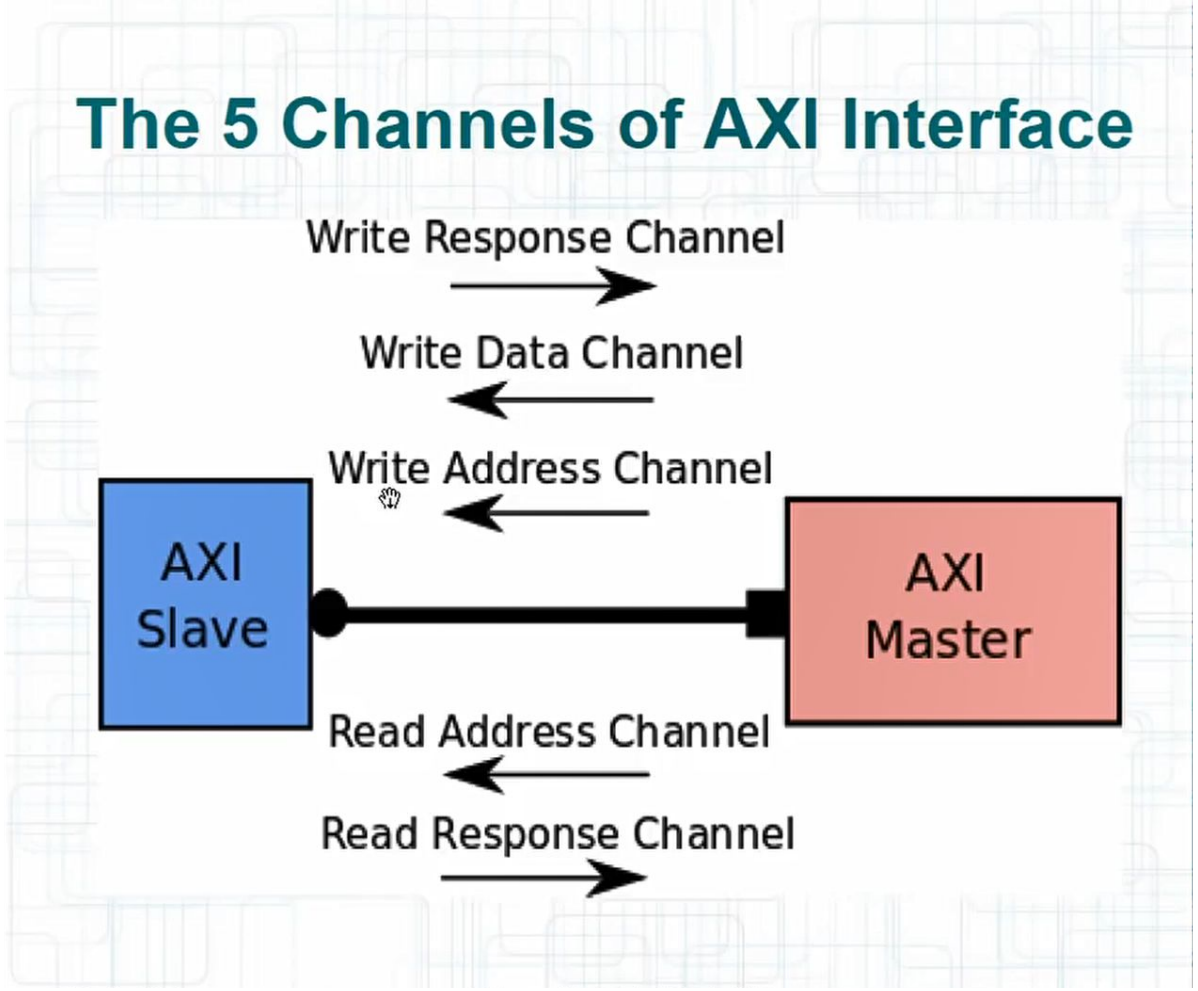
Basic Components in AXI interfaces

1)AXI Master
2)AXI Slave
3)AXI Interconnection

Transaction: Transfering data from one point on chip to another.

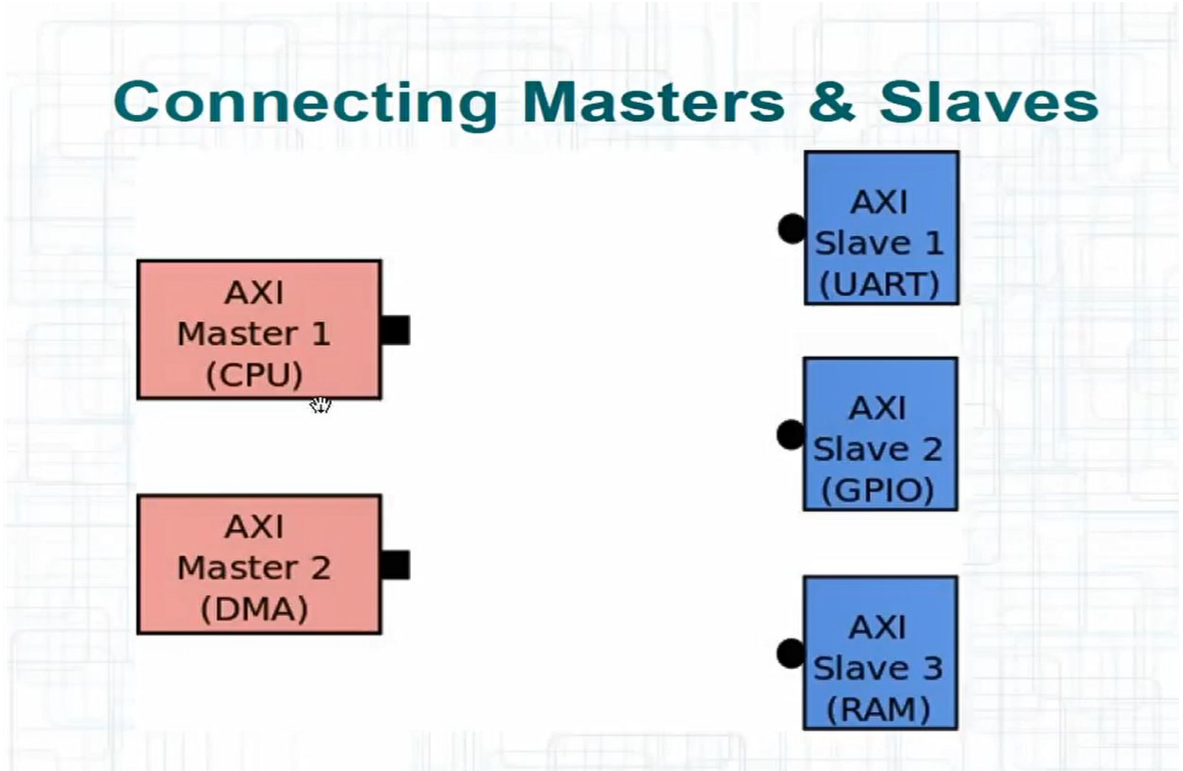


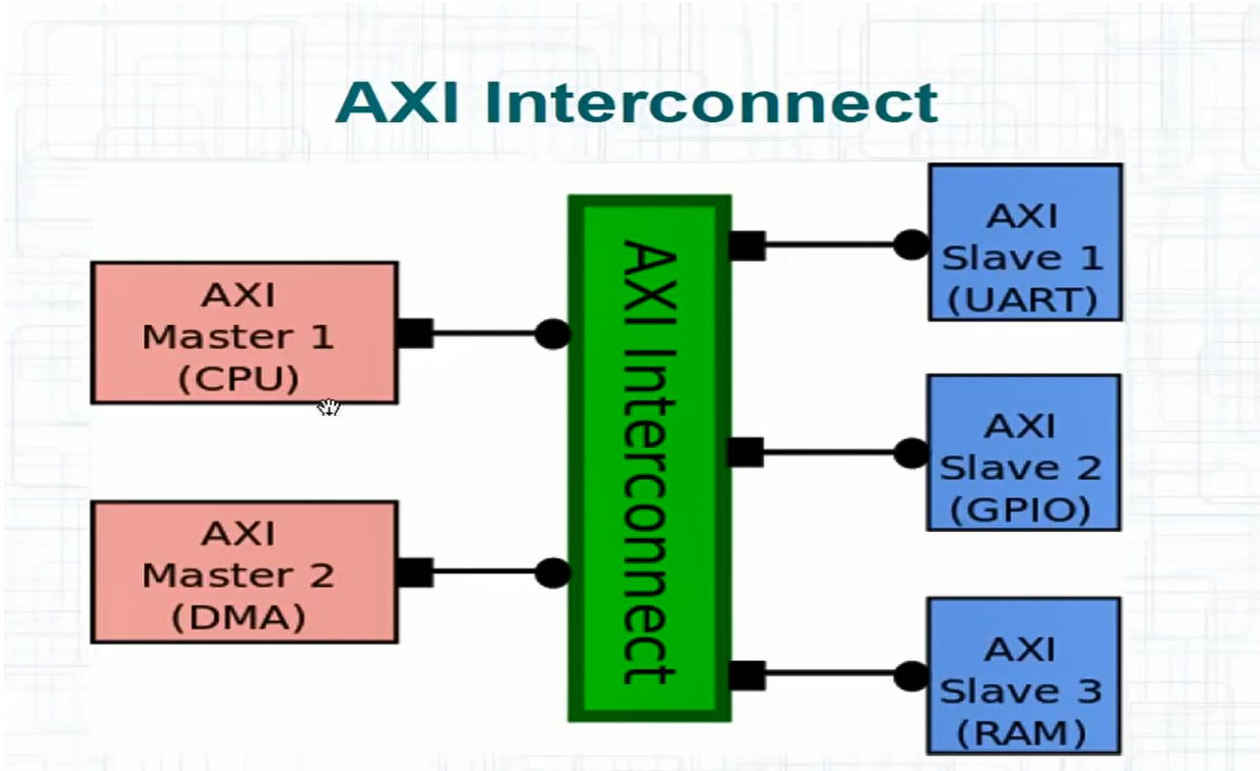
AXI Master: Master triggers to read/write transactions
AXI Slave: Slave responds to masters command



Read Address Channel, master send read address to slave

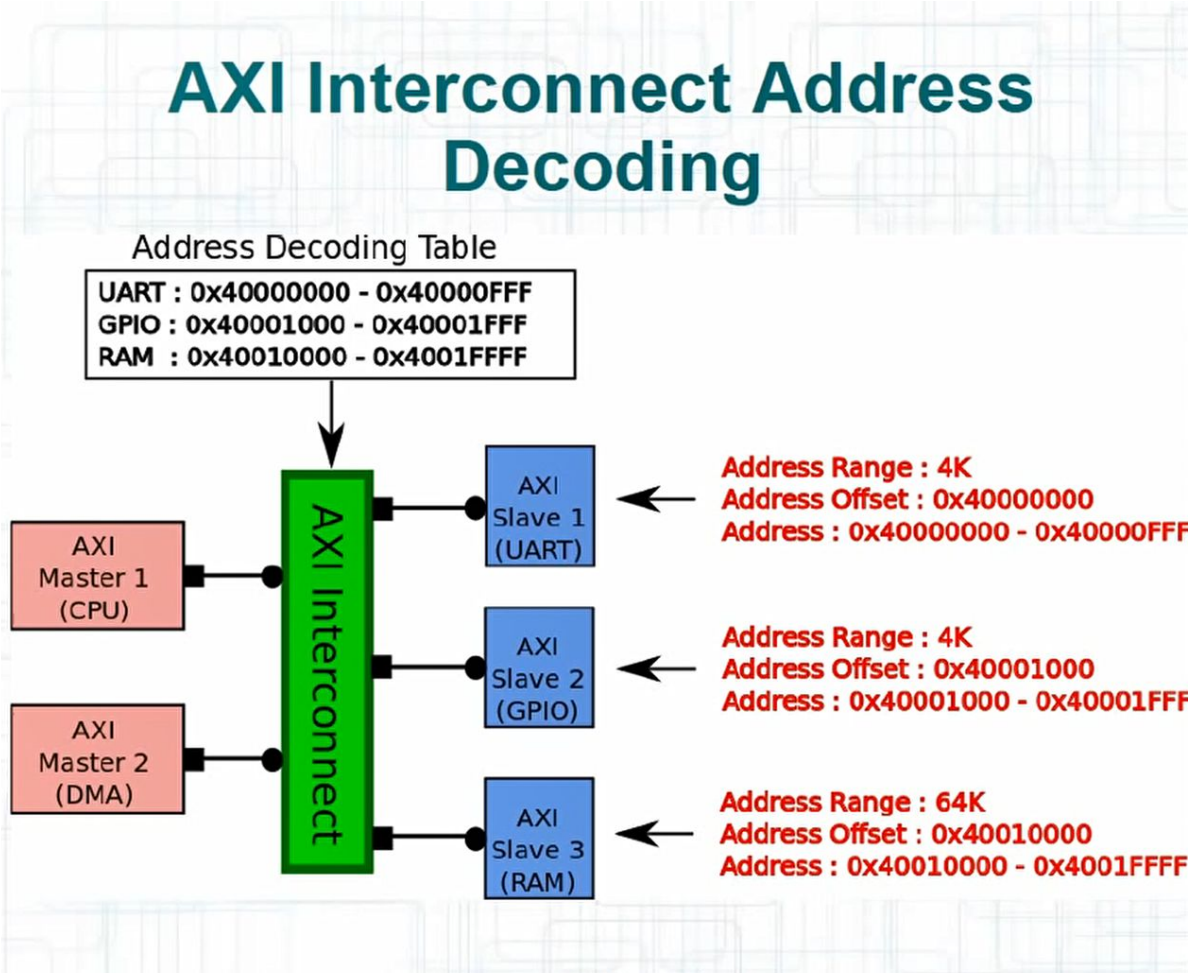
Read Response Channel, slave send to master indicate that read operation has done





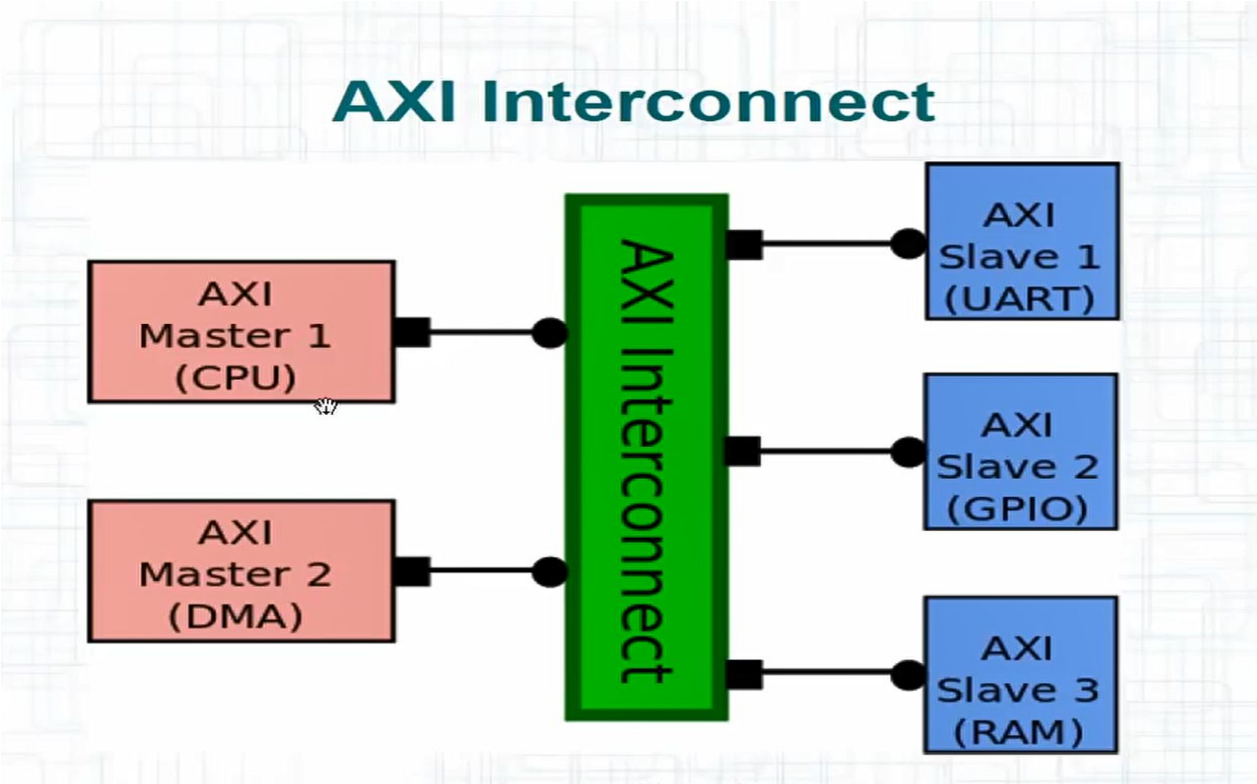
Communcation between multiple masters and slaves cannot be done with direct connection.

Interconnect is slave to master and master to slave.



AXI Interconnect has decoding table which routes to specific slave based on address range

Advantages of AXI Interconnection



1)Multiple masters and slaves communicate

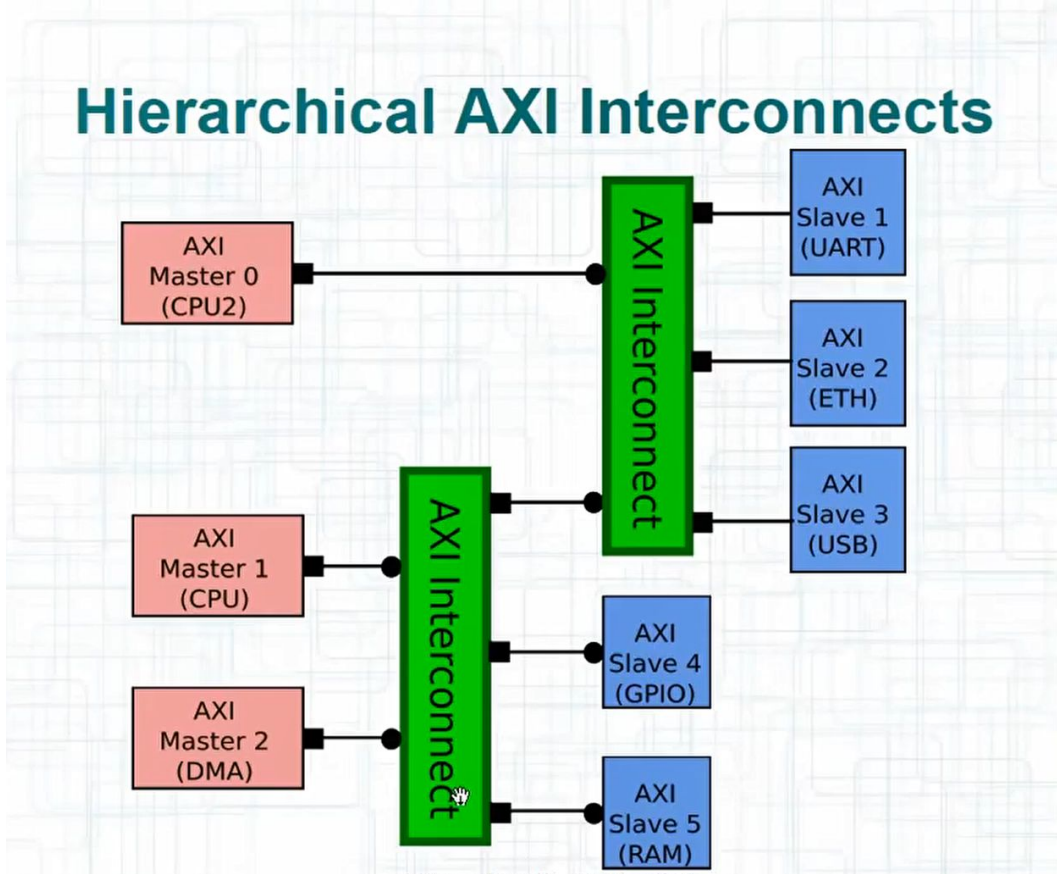
2)Width Conversions

master send 64 bits but if slave capable of receiving only 32 bits then interconnect send two times to slave by sending half of 64 bits each time.

3)Clock domain transformation

master works at 500MHz but if slave works at 250MHz then interconnect send data by frequency division.

4)Interconnect has registers which improve performance

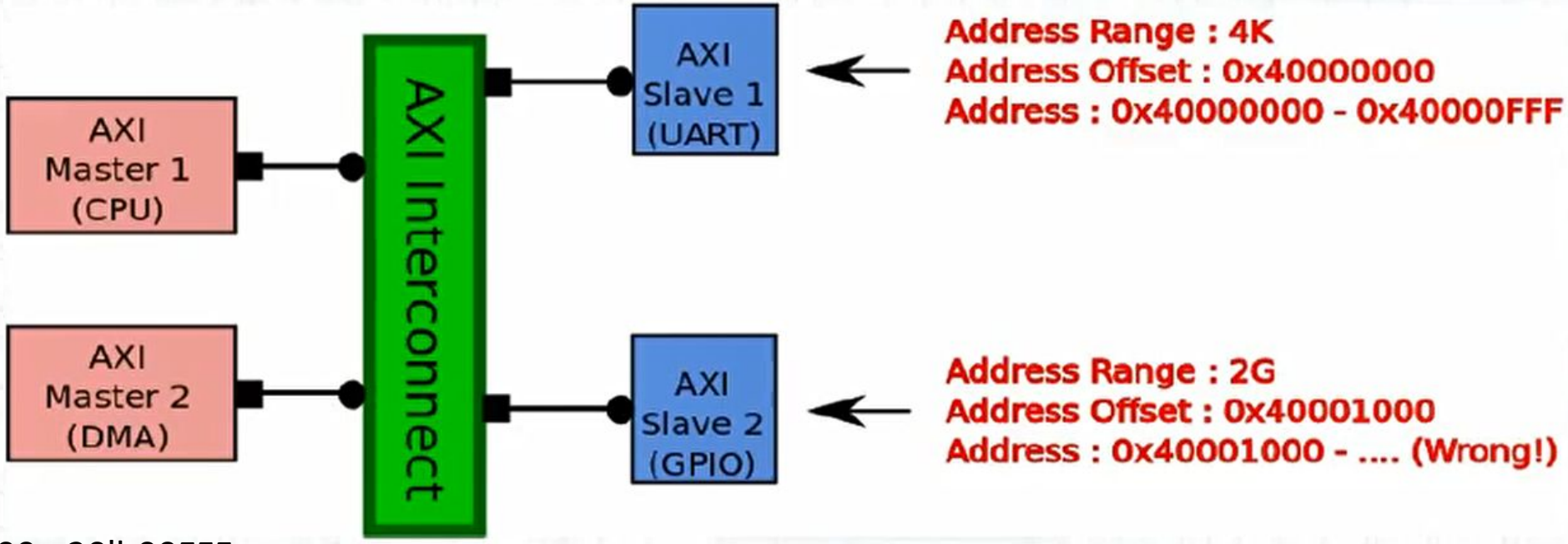


If we have less no. of ports to interconnect than no.of masters or slaves then another interconnect helps to communicate all masters and slaves with others.

This makes architecture to easily expandable.

Some important rule for AXI Interconnection

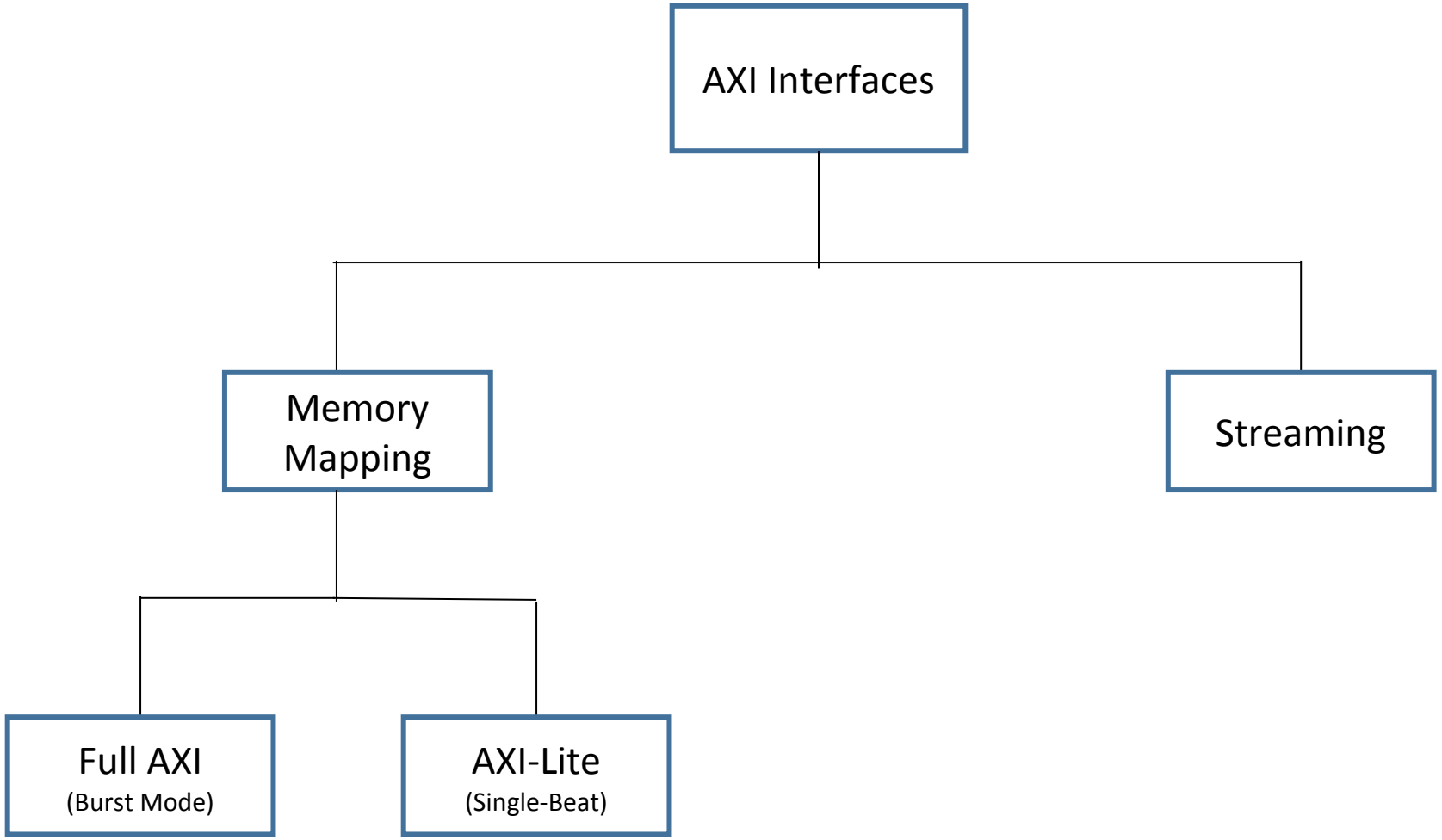
- Meet required alignment in Address



4K:20'h00000 - 20'h00FFF
64K:20'h01000 - WRONG

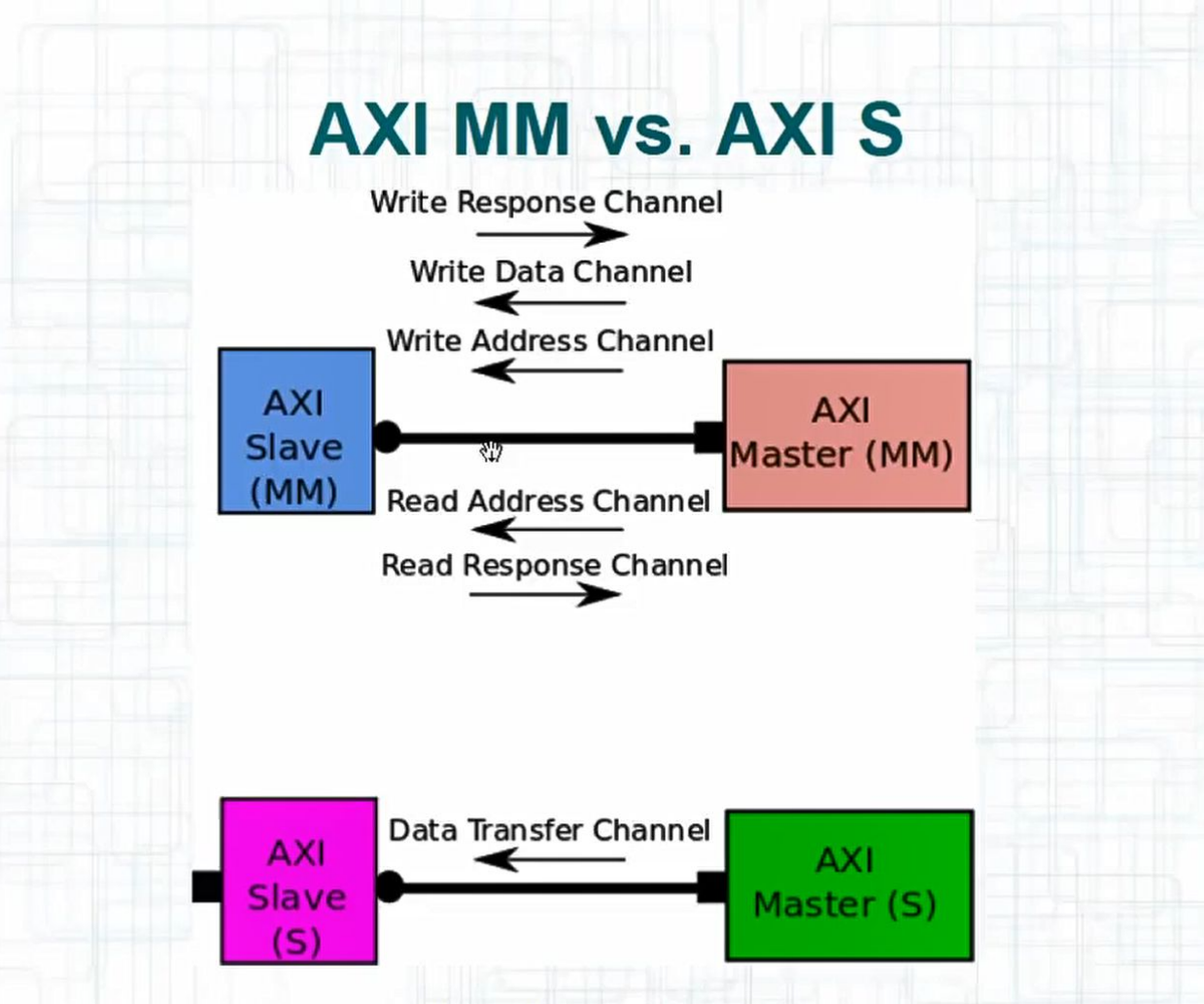
64K:20'h10000 - 20'h1FFFF

Correct : Offset should be in 2G Boundary ! e.g. 0xC0000000


AXI Interfaces
Memory Mapping
Streaming
Full AXI
(Burst Mode)
AXI-Lite
(Single-Beat)

Memory Mapped : Read/write transactions can perform based on destination address

Streaming: Data flow is only in one direction and there is no requirement of address

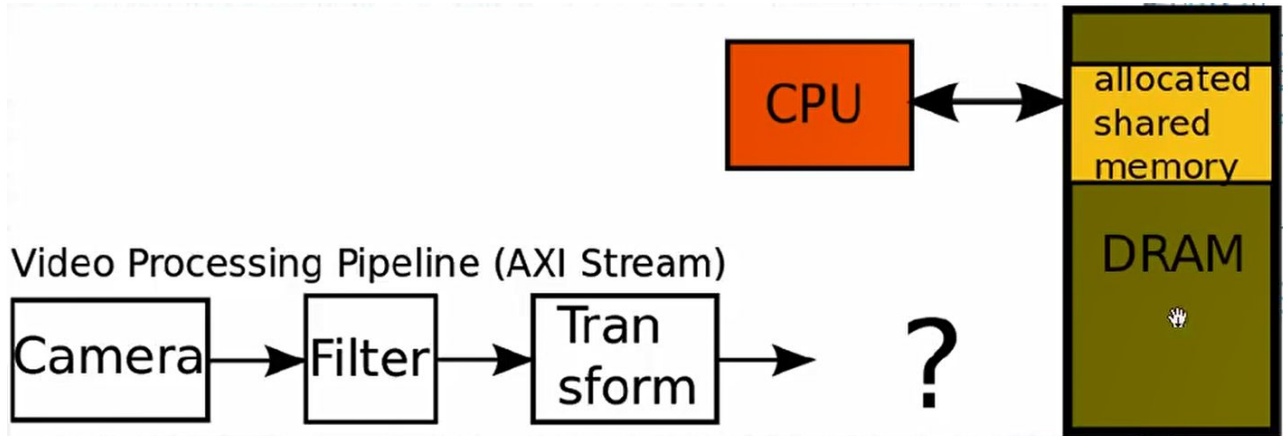


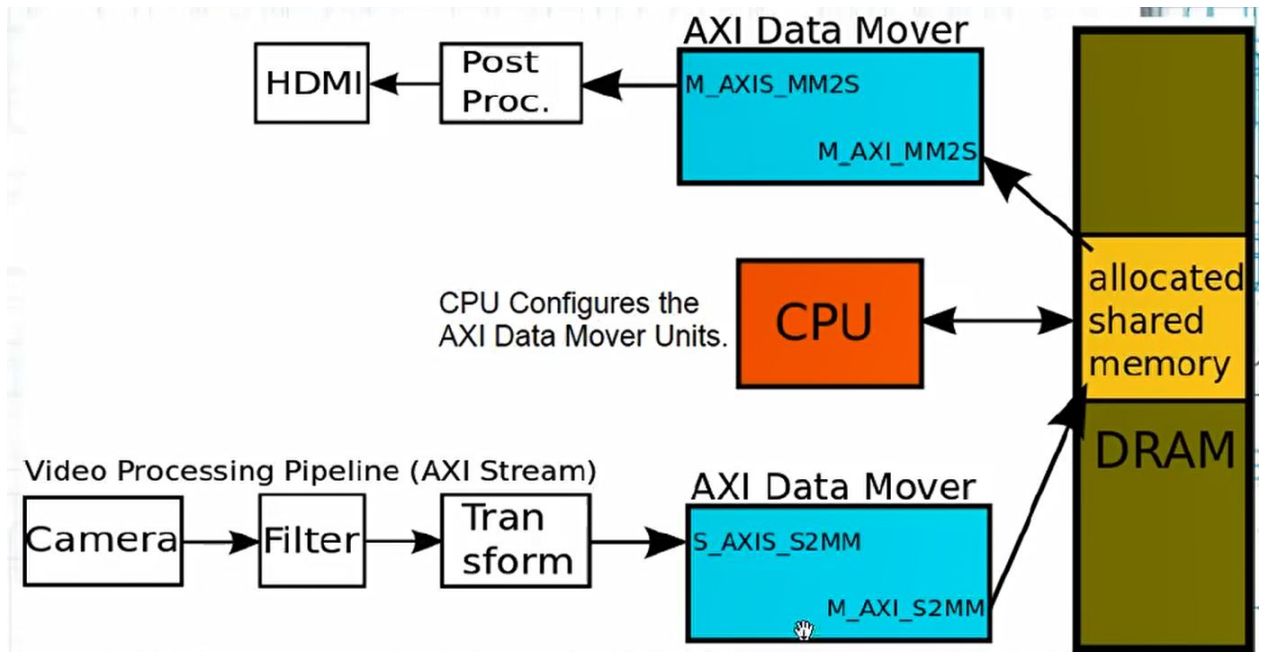
Naming Style of AXI Ports

- Memory Mapped

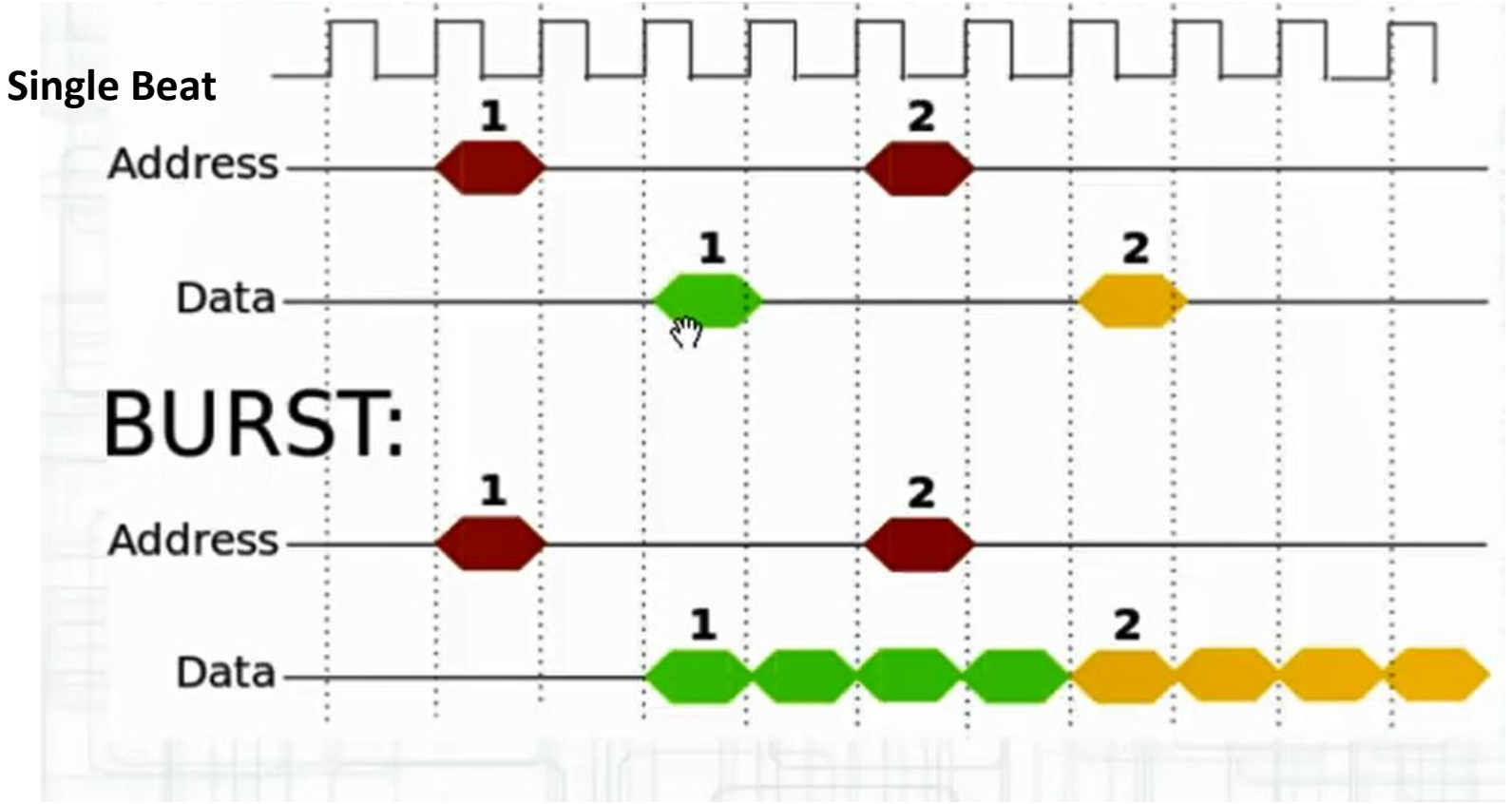
- Stream

- Signal names:
  - e.g. S_AXI_awid, M_AXI_arready ….


CPU
allocated
shared
memory
DRAM
Video Processing Pipeline (AXI Stream)
Camera
Filter
Tran
sform
?


AXI Data Mover
HDMI
Post
Proc.
M_AXIS_MM2S
M_AXI_MM2S
CPU Configures the
AXI Data Mover Units.
CPU
allocated
shared
memory
DRAM
Video Processing Pipeline (AXI Stream)
Camera
Filter
Tran
sform
AXI Data Mover
S_AXIS_S2MM
M_AXI_S2MM

Full AXI vs AXI-Lite



For one call of transaction of AXI-Lite only one chunk of memory has transferred.This chunk of data can be more than one bit like 16-bit also.

For one call of transaction of Full AXI multiple chunk of memory can transfer.

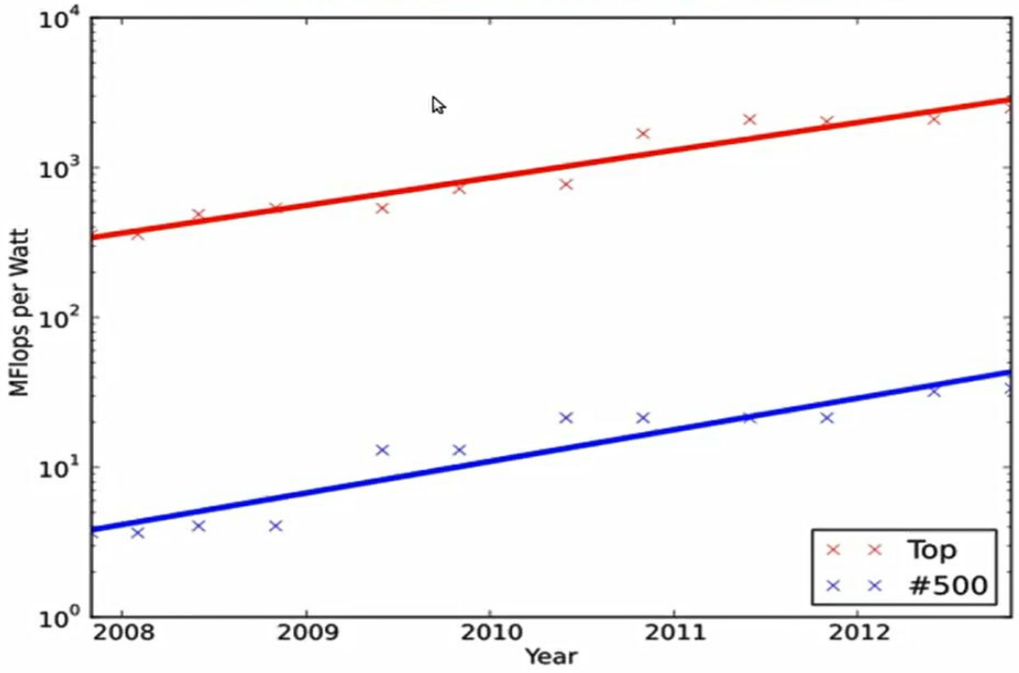
ZYNQ Architecture

1951 UNIVAC I : **0.015** operations per **1** watt-second

Half a century later!

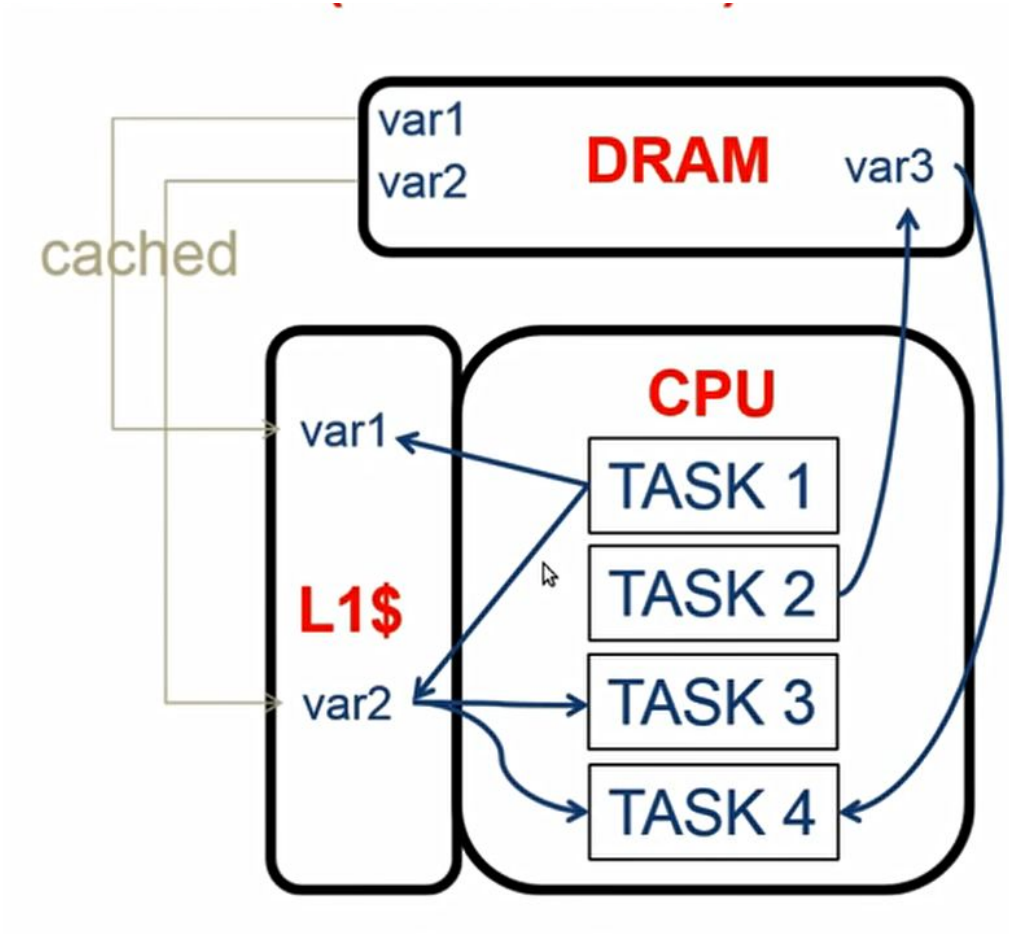
2012 **ST P2012** : **40** billion operations per **1** watt-second

Performance Per Watt!!



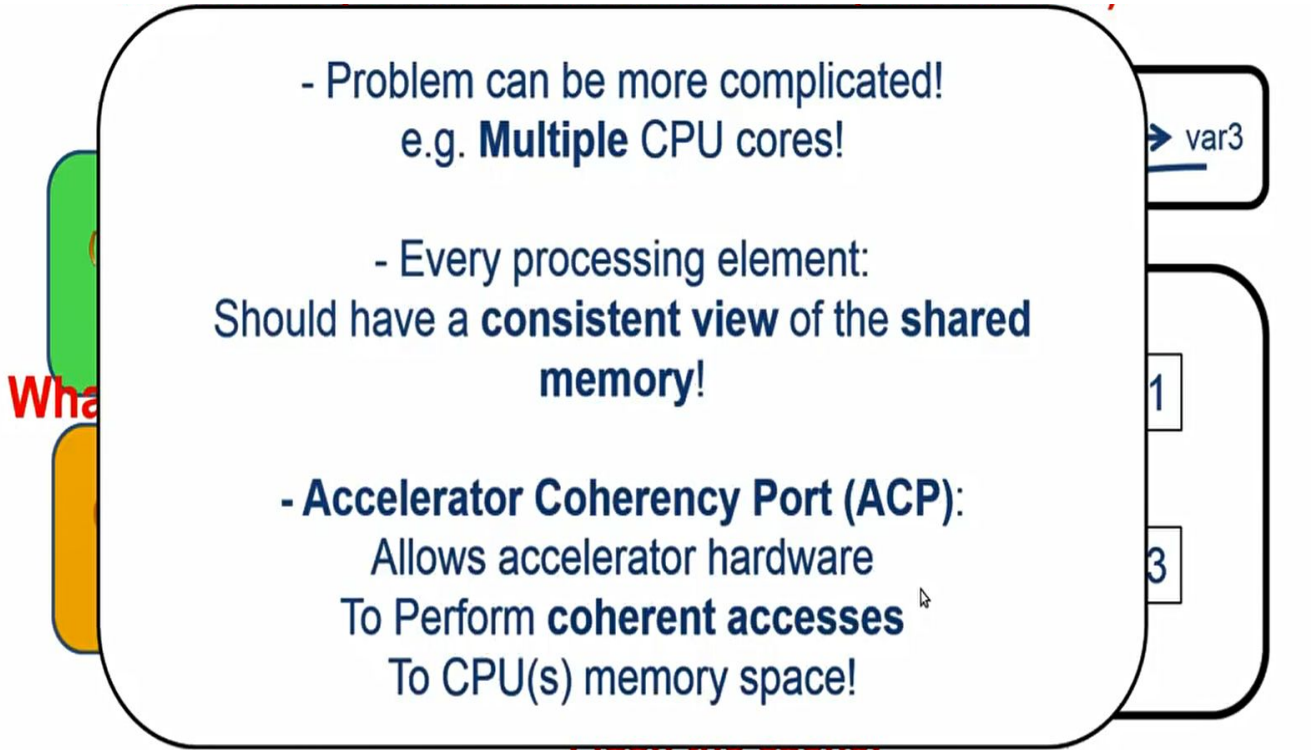
Solution : Specialized functional units (Accelerators)

What really Accelerators does???



Traditional CPU

Modern CPU



support variables overhead. How to access them ?

FPGA vs CPU

FPGA

1)FPGA is extensively used for processing large size of data and return it like FFT .
2)High I/O applications so that interfaces many peripherals.
3)Massively parallel algorthims, performing a task through performing its subtasks at a time to complete the task.

CPU

FPGAs are not capable efficiently to handle many flow control operations like if-else conditions,for loops. For image compression techniques like DCT FPGA is not flexible.In such cases CPU performs very well.

Implement CPU on FPGA

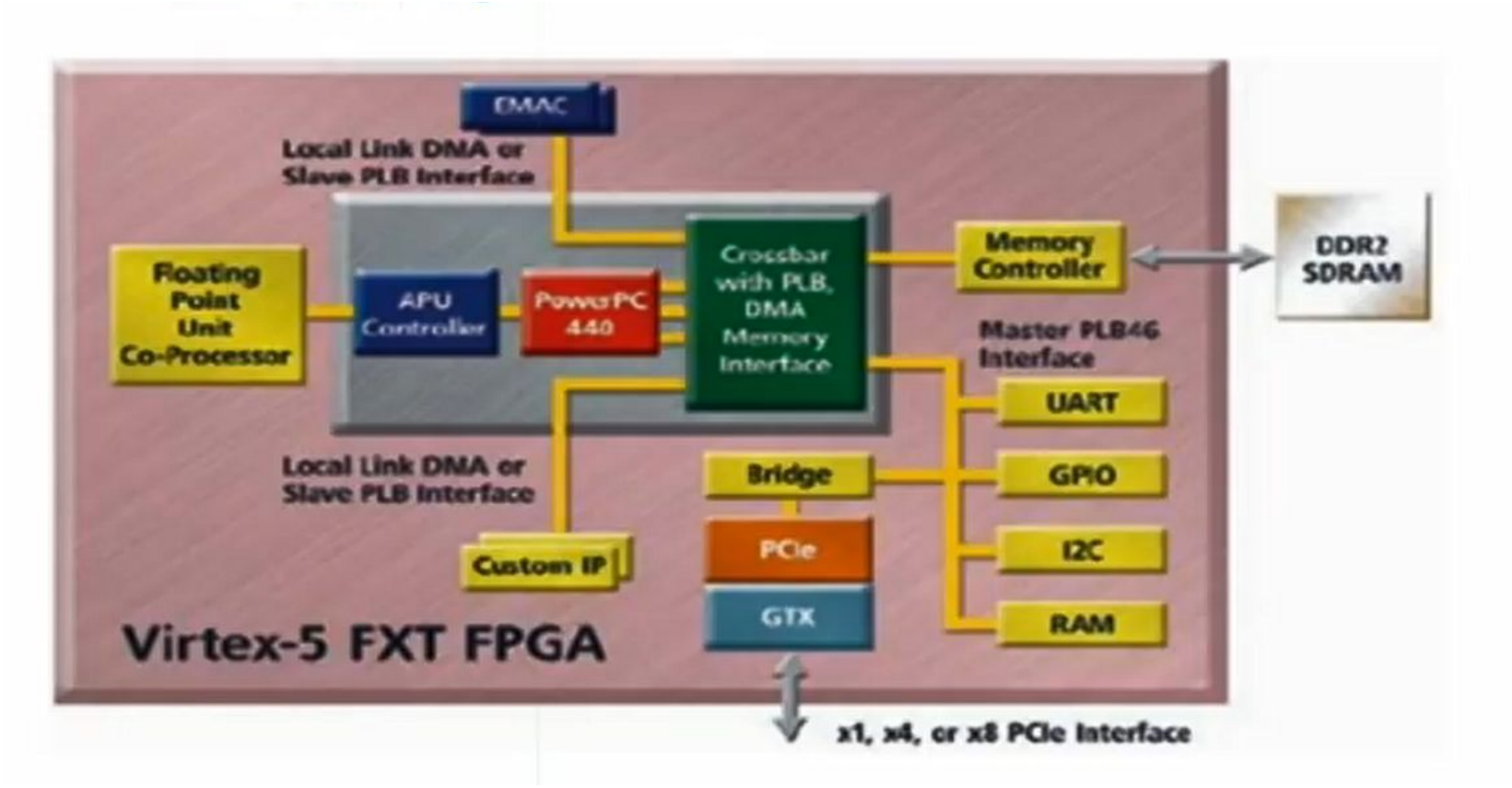
MicroBlaze

Xillinx has designed a microprocessor core MicroBlaze which has CPU on FPGA.

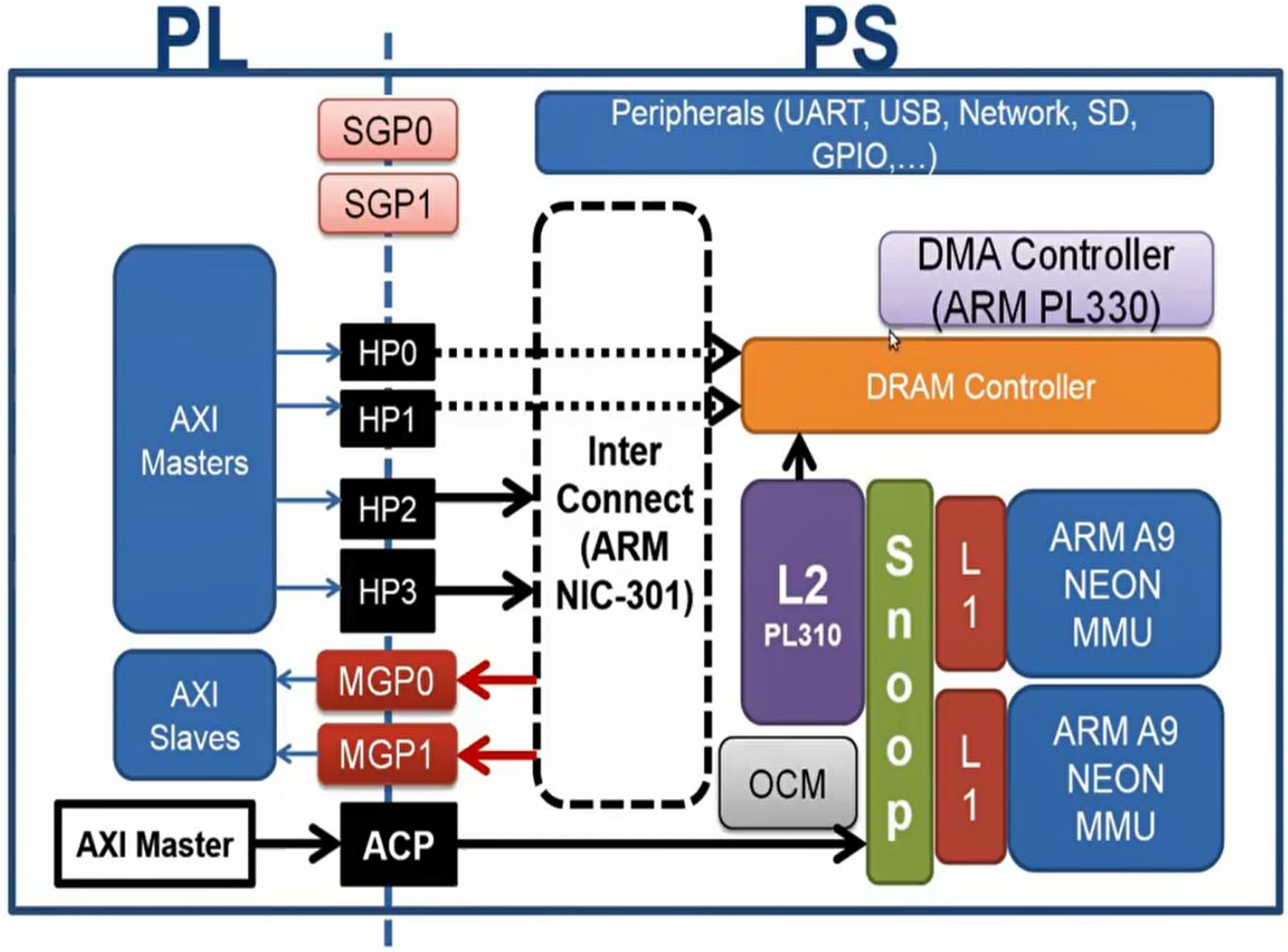
But this have many disadvantages,
1)It occupies more area on FPGA
2)The CPU speed is not at all good as an individual CPU

Viertex-5 FXT FPGA

This FPGA has CPU core IP block on FPGA this designed by ARM and deployed in ZYNQ architecture



Xillinx ZYNQ Architecture



This bus is through which PS control modules in PL are MGP0 MGP1 An Interconnect to communicate between all these modules and to masters and slaves in PL. turn this AXI master connects modules in PL so that communicate between PL and PS. This bus is through which PS control modules in PL are MGP0 MGP1 An Interconnect to communicate between all these modules and to masters and slaves in PL. turn this AXI master connects modules in PL so that communicate between PL and PS. ort is of size 64-bits.In

1)Programmable Logic(PL):DSP slices ,LUTs,FFs,Clock Managers,ACP Transivers,ADC or DAC this part resembles FPGA.

2)Programmable System(PS):Two ARM CPU IPs,Caches,Interconnect,DRAM.AXI DMA controller,Peripherals,OCM.

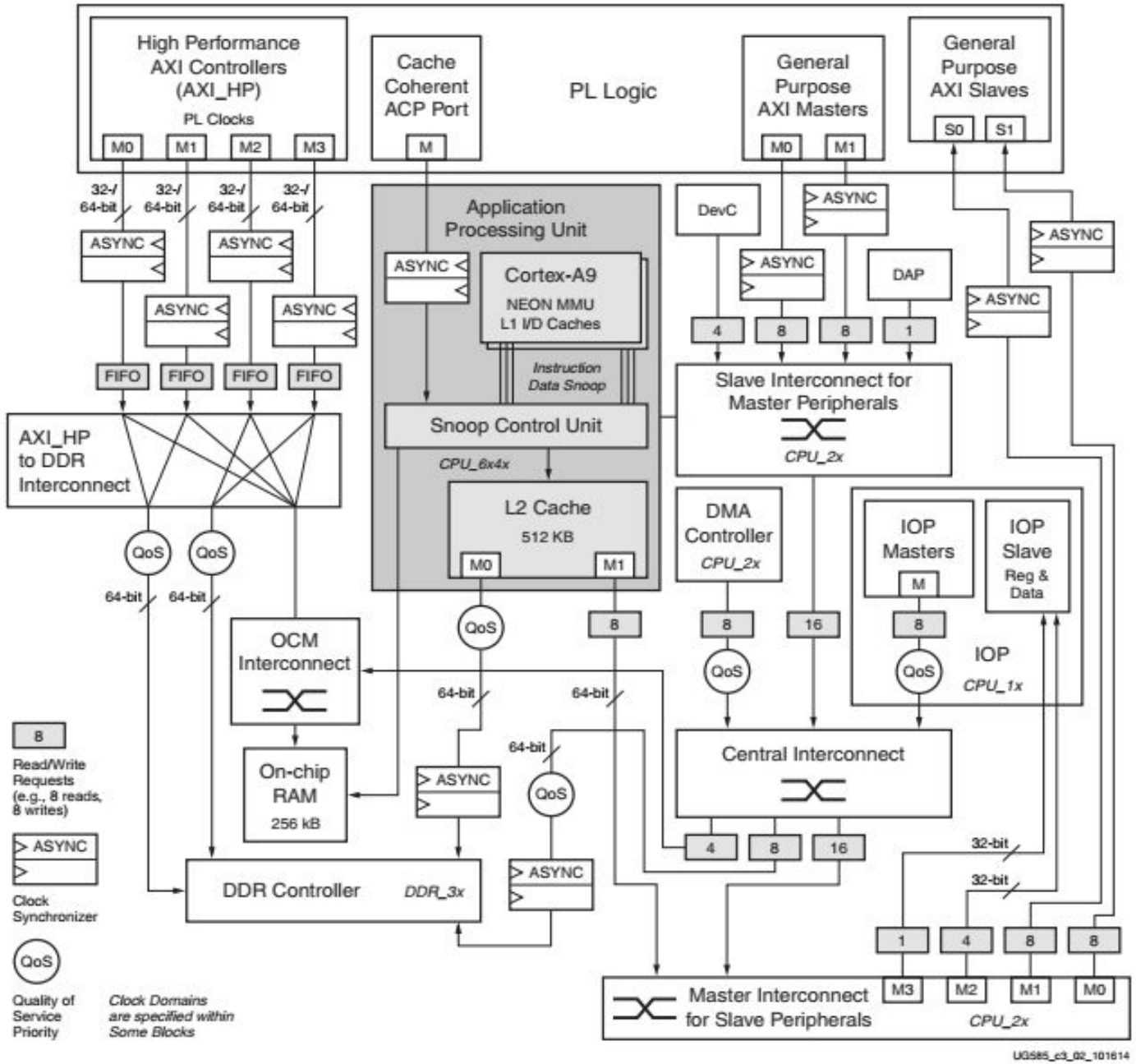


Figure 3-2: **APU System View Diagram**

Table 4-1: **System-Level Address Map**

Address Range	CPUs and ACP	AXI_HP	Other Bus Masters[1]	Notes
0000_0000 to 0003_FFFF[2]	OCM	OCM	OCM	Address not filtered by SCU and OCM is mapped low
	DDR	OCM	OCM	Address filtered by SCU and OCM is mapped low
	DDR			Address filtered by SCU and OCM is not mapped low
				Address not filtered by SCU and OCM is not mapped low
0004_0000 to 0007_FFFF	DDR			Address filtered by SCU
				Address not filtered by SCU
0008_0000 to 000F_FFFF	DDR	DDR	DDR	Address filtered by SCU
		DDR	DDR	Address not filtered by SCU[3]
0010_0000 to 3FFF_FFFF	DDR	DDR	DDR	Accessible to all interconnect masters
4000_0000 to 7FFF_FFFF	PL		PL	General Purpose Port #0 to the PL, M_AXI_GP0
8000_0000 to BFFF_FFFF	PL		PL	General Purpose Port #1 to the PL, M_AXI_GP1
E000_0000 to E02F_FFFF	IOP		IOP	I/O Peripheral registers, see Table 4-6
E100_0000 to E5FF_FFFF	SMC		SMC	SMC Memories, see Table 4-5
F800_0000 to F800_0BFF	SLCR		SLCR	SLCR registers, see Table 4-3
F800_1000 to F880_FFFF	PS		PS	PS System registers, see Table 4-7
F890_0000 to F8F0_2FFF	CPU			CPU Private registers, see Table 4-4
FC00_0000 to FDFF_FFFF[4]	Quad-SPI		Quad-SPI	Quad-SPI linear address for linear mode
FFFC_0000 to FFFF_FFFF[2]	OCM	OCM	OCM	OCM is mapped high
				OCM is not mapped high

Address range is reserved for each module.

On the ZYNQ boards we can run set of Operating Systems(OS)

1)Bare Metal

This is used for running a specific application without any concurancy between several applications.This makes not to consume power for other processing tasks.This is the simplest option and its efficient.

2)Linux

On the ARM CPU we can run Linux.By this system can gain lot of possiblities which are very difficult to obtain through bare metal.
Petalinux tools unables to develop xillinx embedded systems.